Optimization-based Rate Control in Overlay Multicast

Zhang Lin

A Thesis Submitted in Partial Fulfillment of the Requirements for the

Degree of Master of Philosophy in

Information Engineering

The Chinese University of Hong Kong

June 2004

Abstract

Congestion control for IP multicast on the Internet has been one of the main issues that challenge a rapid deployment of IP multicast. People have proposed various kinds of schemes, among which the optimization-based rate control schemes for multi-rate multicast is our main concern in this thesis. In such schemes, an optimization problem is formulated with the object being maximizing the aggregate utilities of receivers under network link capacity constraints. However, in the solution approach proposed for this problem, network nodes are assumed to be capable of measuring flow rates in links, computing and exchanging information, none of which actually exists in current Internet. However, in overlay multicast, all these tasks can be performed at participating end hosts with change needed in the underlying network infrastructure.

In this thesis we tackle the problem of rate control in overlay multicast, which poses some new challenges compared to IP multicast. First, some end hosts relay traffic for their child hosts in the tree, so the traffic rate at a host cannot exceed its father's rate. This poses a special constraint to the optimization problem. Second, rate control is achieved through layered multicast in previous work, in which data is encoded into multiple layers and receivers choose from a discrete set of rates by receiving a subset of these layers. In overlay multicast, end hosts can be provisioned with various kinds of end-to-end rate adaptation techniques, so receivers can choose from a continuous range of rate values.

Based on previous work, we propose two distributed algorithms (primal-based and dual-based) to solve the above problem in overlay multicast. They both prove to converge to the optimal solution through iterative steps. For practical implementation,

we design protocols for end hosts to control overlay multicast rates. Through simulations, we show the convergence properties of the algorithms in distributed network environment measure some overhead during implementation and show their scalability and efficiency.

摘要

擁塞控制是阻礙IP組播快速應用的一個主要障礙。近年來，應用于多速率組播的基於最優化的速率控制機制得到了廣泛的研究。在這種機制中，擁塞控制可以被認爲是一個基於網絡鏈路本身帶寬限制，使所有用戶的效用達到最大的最優化問題。目前的一些解決方案都假設網絡鏈路可以測量本身的流量，計算一些信息量以及與其他鏈路互相交換這些信息，然而現實中的網絡並非如此。我們的方法是利用應用層組播技術，所有任務都可以由其中的主機來執行，從而不需要對現有網絡做任何改變。

本文我們在原有的最優化模型上研究了基於應用層組播的速率控制問題。首先，組播樹中的一些主機節點要轉發數據包給子節點，所以一個節點的接受速率不能大於父節點，這就給該最優化問題增加了一個新的限制。其次，之前的工作中速率控制是通過分層組播來實現的，其中數據被編碼成若干層，接受者通過接受其中的某些層達到不同的速率，然而在應用層組播中，利用適用于主機的端對端速率調适技術，接受者可以在連續的區間內選擇自己的速率。

我們提出了兩种分佈式算法（primal-based和dual-based）以解決上述問題。這兩种類型的算法都可以通過迭代的方式收斂到問題的局部最優解。另外，考慮到實際的應用問題，我們設計了相應的協議用於應用層組播的速率控制。仿真試驗充分驗證了算法在分佈式環境中的收斂性,對協議實現時帶來的額外開銷的測量亦表明他們擁有良好的擴展性。

Acknowledgement

My foremost thank goes to my thesis adviser Prof. Tak-Shing Yum, Peter for his guidance, support and friendship. Without him, this thesis would not have been possible. I thank him for his patience and encouragement that carried me on through difficult times, and for his insights and suggestions that helped to shape my research skills. If I am half as good as he, I will consider myself very successful.

I am grateful to all my friends who have given me encouragement and help in both study and life during these days. They are: Zhang Xinyan, Tao Dacheng, Wang Wei, Deng Ning, Xiao Lurong. Thanks to you all, without you, my life here cannot be so colorful and my research cannot go so smoothly.

Last but not least, I thank my dearest parents and boyfriend for always being there when I needed them most, and for supporting me through all these years.

Contents

Chapter 1 Introduction

1.1 Why use economic models?

With the advances in computing and networking technology, thousands of computers can be interconnected to provide a large collection of computing and communication resources. Meanwhile, a growing number of users are using such systems to obtain various kinds of services. Due to the heterogeneity in applications and users, the distributed computer systems reveal the complexity in the organization and management of the resources and services they provide.

This massive complexity makes traditional approaches to resource allocation impractical in modern distributed systems. Most resource sharing algorithms proposed are characterized by at least one of the two common features: *centralization* and *consensus*. Such algorithms attempt to allocate resources that optimize some system-wide performance metric (e.g. minimize average delay; maximize total throughput [1], etc). Seeing the complexity described above, it's difficult to define a single system-wide performance metric, and what's more, improving global system performance is often in conflict with individual user's satisfaction. Centralized or consensus based algorithms are usually impractical in a dynamic system owned by multiple organizations. Thus people were led to rethink the problem from another point of view --- try to find similarities between computer systems and economics. Microeconomics provides a set of tools for the study of resource sharing algorithms

[2][3][4]. Particularly, the introduction of *utility* makes multi-objective optimization possible. So we need not bother with the problem of finding a single system-wide objective.

In the economic model we use in our work, consumers are the users of the applications. The system is the primary supplier and controls resources like communication links shared by all consumers. Supplier controls access to its resources via prices, and consumers buy resources from the supplier to satisfy their service requirements. Prices are adjusted by the supplier based on the demand of all consumers. Through the interaction between consumer and supplier, a global equilibrium can be reached.

1.2 Why Overlay?

Many present day real-time applications, like teleconferencing and audio/video streaming, require communication within a group, with multicast an often requirement. Optimal rate control for resource allocation in IP network has been studied by many people [5, 6, 7, 8]. A most common formulation is to maximize the aggregated utilities of all users, subject to multiple constraints. Utility maximization is a more general formulation because resource allocation with the fairness property is utility maximizing when the utility has a special form. Utility is a concept borrowed from economics, which could be a measure of say, the perceived quality of audio/video, the user satisfaction, and the amount paid by the receiver.

The work of optimal rate control in IP multicast [5, 6] mainly deals with multi-rate multicast system architecture. In conventional multicast, all users in the same multicast group receive service at the same rate. However, due to the varying characteristic among different users, multi-rate multicast is proposed to allow users to receive service at different rates. In this way, receiver is allowed to receive data at a rate that is commensurate with its requirements and capabilities, and also with the capacity of the path leading to it from the source.

In IP multicast, multi-rate transmission can be attained by hierarchically encoding real time signals. In this approach, a signal is encoded into a number of layers that can be incrementally combined to provide progressive refinement [9]. Every layer is transmitted as a separate multicast group and receivers adapt to congestion by joining and leaving these groups. Refer to [9] and [10] for internet protocols for adding and dropping layers.

In recent years, overlay multicast has attracted lots of attention [14, 15, 16, 17, 18]. We choose to do rate control on it since Overlay Multicast has the potential to address most problems associated with IP Multicast.

1. IP multicast has seen slow commercial deployment by ISPs and carriers. This is caused mainly by its deficiency in infrastructure support [19]. But in overlay multicast, since all packets are transmitted as unicast packets and no support for multicast is needed in underlying network, deployment can be accelerated.
2. In IP multi-rate multicast, the receiver selects proper layers to achieve its desired service. Thus it can only select from a bounded discrete rate set. In the previous work, when the resultant computed rate is not in the set, it has to be rounded to the nearest set element. In overlay case, rate allocation is achieved by the coordination of end hosts. Data relay happens on end hosts, on which various functionalities can be co-located. Due to the flexibility of end host, all end-to-end stream adaptation techniques (frame dropping, transcoding, etc.) can be applied. Therefore, the receiver can select its rate from a continuous range, which is more precise than the discrete case of IP layer.

1.3 Our Contribution

The optimization-based rate control in IP unicast and multicast has a number of approaches. We select two most popular algorithms of them, named primal-based and dual-based, as the basis of our work. They are both scalable, distributed and proved

to converge to the optimal point where the objective function is satisfied in previous work. We apply these two algorithms into our work and show their viability in overlay multicast environment.

From the literature, the problem in network-layer (either unicast or multicast) is formulated as a non-linear optimization problem [20]. The objective is to maximize the aggregated users' utilities. One constraint arises from the fact that the traffic offered to a network cannot exceed the capacity limits the network can carry. Another constraint concerns the user side. Each user has an acceptable range of received rate and the resulting rate cannot surpass such a domain.

Aside from the basic framework described above, the problem in overlay multicast has some specific characteristics to consider. So although using similar algorithms, we are still facing challenges to implement such a scheme on overlay multicast. And the challenges make this problem a totally different one which cannot be solved by any current solution. Below we address the challenges and propose our strategies.

One challenge is that current solutions in IP multicast all assume that network link can measure its available bandwidth, adjust link price according to the congestion level, and then transmit updated price to users. Therefore, those routers attached to the links need corresponding modifications to implement all these functions. In this way, current solutions cannot be applied to network without causing changes to network infrastructure. That is why the idea of using overlay multicast comes. Overlay network is organized and operated totally by end hosts and doesn't need change anything to existing infrastructure. Any functions once assigned to links are now migrated into end hosts. This is what we do—to propose an end-host-based solution, and design protocols to implement it. Different from previous solution, our solution can be easily applied onto overlay multicast with high flexibility.

Another challenge is an additional constraint brought by overlay multicast's own characteristic. In overlay multicast, some end host acts as both a receiver and a sender.

These hosts help others in relaying their traffic. Intuitively, a host cannot send data at a rate higher than the one it received, so this adds a data rate constraint in the original optimization problem formulation. Correspondingly, we propose two distributed algorithms and design protocols to apply on overlay multicast. In the algorithms, the hosts who gain help from other hosts share some link costs with their helpers or pay the helpers directly. Such actions arise from the data rate constraint in problem formulation. We prove that the rate control process converges to the optimal point, at which the aggregate utility of all receivers is maximized. We will cover the algorithm details in later part.

After considering all difference between overlay multicast and IP multicast, we propose new protocols to realize optimal rate allocation in overlay multicast. These protocols need not change the network infrastructure and can be flexibly deployed at participating end hosts. Also we evaluate the two protocols in simulations, thus people can select one of them after investigating the performance factors they concern in application.

1.4 Thesis Organization

The remainder of the thesis is organized as follows.

In Chap 2 we discuss the related work on overlay multicast, IP multicast congestion control schemes and optimization-based rate control in IP unicast and multicast.

Chap 3 introduces the problem of rate control for optimal resource allocation in overlay multicast. We present our problem formally as an optimization problem. Also we state the algorithm requirements and present a primal-based algorithm and a dual-based algorithm for it. Afterwards we give the convergence analysis for these two iterative algorithms.

Chap 4 describes how the algorithms can be implemented in a real overlay network. We design the protocols and demonstrate their convergence through simulations in

asynchronous network environment. Simulation results show that our proposed scheme is scalable, flexible and easy to implement without bringing much overhead into current network infrastructure.

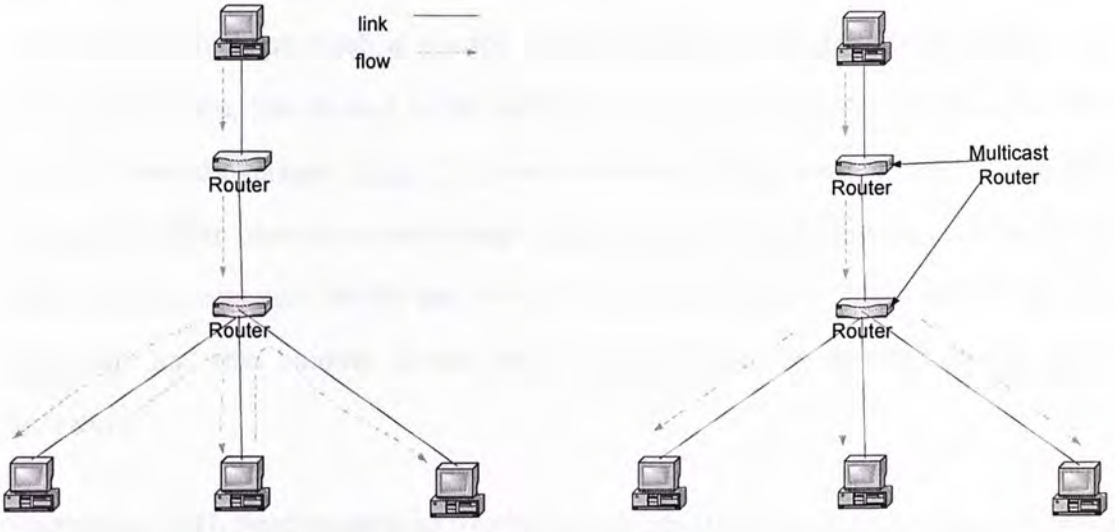
Chap 5 gives a summary of my thesis.

Chapter 2 Related Works

In this chapter, first we briefly introduce overlay multicast which is the basic framework of our work. Then we survey various schemes proposed in the literature in the last decade in the area of congestion control especially for multicast video applications. Schemes presented in this part are mainly for single-source multicast case. Finally we discuss in detail the optimization-based rate control for multi-rate multicast, which is of our most interest in this thesis.

2.1 Overlay Multicast

Overlay Multicast has recently become a hot topic [14, 15, 16, 17, 18] as an alternative to IP multicast. Overlay multicast uses current Internet as the low level infrastructure to provide multicast services to end hosts. By running certain distributed algorithms, participants of a multicast session organize themselves into an overlay network. All communications are then carried out through unicast connections between nodes in the overlay. This offers the advantage of possible immediate deployment since it can utilize all the flow/congestion control facilities available in unicast schemes. Also since overlay multicast is built on the application layer, it can provide significant flexibility to satisfy heterogeneous application demands. And the disadvantages exist in higher delays and inefficient use of network bandwidth. Some packets may traverse the same link back and forth, as illustrated in Figure 2.1. Overlay multicast is also referred to as end-system multicast or application-level multicast.



a) Application-layer Multicast **b) Network-layer Multicast**

Figure 2.1 Multicast at different layers

Existing overlay multicast schemes can be classified into two categories according to their structures: *end-to-end* and *proxy-based*. In end-to-end overlay, each participant of the multicast session shares the responsibility to forward data to other participants. End hosts self-organize into a multicast tree. Narada [15], Yoid [16] and ALMI [17] are examples of such a structure. Using proxy-based overlay structure, Scattercast [18] and Overcast [14] form a hierarchical structure compared to the flat one in end-to-end overlay structure. In this structure, the overlay multicast network provides services through a set of distributed proxy nodes, which communicate with hosts and with each other using standard unicast mechanisms. Also these proxy nodes forward and replicate data packets on behalf of the senders. In both cases, multicast node is defined as the member in the multicast tree.

Here we present four representative schemes of overlay multicast to get a quick view: Scttercast, Overcast, Narada and ALMI. Narada and Scattercast intend to minimize delay for each member, while Overcast wants to maximize the available bandwidth for each member. ALMI strives to minimize the system cost defined by certain application-specific performance metric. Narada and Scattercast use a *mesh-first*

approach, in which group members are connected in a mesh first and then the multicast tree is built on top of this mesh. Overcast and ALMI use a *tree-first* approach in which no mesh is needed and the multicast tree is formed directly. As mentioned before, Narada and ALMI belong to end-to-end overlays and the other two are proxy-based overlays. Table 2.1 gives a summary of the comparison of these four schemes in their objectives and design approaches. In the following, we introduce these four projects very briefly and concentrate on issues about mesh initialization if applicable and tree buiding. Some other issues will not be detailed due to space limitation.

Scattercast [18]. Scattercast is an overlay proxy-based multicast architecture. When a new node joins the multicast session, it bootstraps itself via a well-known list of rendezvous points and then relies on the gossip-style discovery algorithm to locate other members. When the new node encounters other members who have already been in the session, it selects some of them as its neighbors if they satisfy the degree constrains. The initial mesh is randomly formed. When performing optimization, latency is the primary metric considered, and based on which, the member decides to accept others as neighbors or to change neighbors according to a pre-defined cost function and threshold. A distance vector routing protocol (DVMRP) [22] is running on top of the mesh to build the tree. The routing metric used in the DVMRP is also the latency between neighbors.

Narada [15]. Narada intends to serve small size and sparse groups such as audio/video conferencing. Both Narada and Scattercast use mesh-first approach with different mesh optimization methods. Narada assumes that the new node is able to get a list of current group members by some rendezvous points. It randomly chooses some members as its neighbors under degree constraint. The join process succeeds when at least one of these members accept the new member as its neighbor. After joining the mesh, the new member exchanges messages with its neighbors to learn information about other members. Every member periodically evaluates the utility of

adding a link to another member and deleting existing links, and makes corresponding decisions based on defined thresholds. The way of routing and building multicast tree is the same as Scattercast.

Overcast [14]. Overcast uses *tree-first* approach in building multicast trees. The objective is to maximize the available bandwidth to the source for each member, potentially at the expense of increased delay. After initial contact, a new node tries to locate itself further away from the source without sacrificing available bandwidth to it. Periodically, the node evaluates the bandwidth to the source through its parent as well as through its siblings. If the bandwidth through any of the siblings is about as high as the one through the parent, that sibling will be chosen as the new parent for this node. In this way, the node will be located as far from the root as possible without losing bandwidth.

ALMI [17]. ALMI is designed to minimize the cost of the distribution tree. The distribution tree in ALMI is formed as a Steiner Minimum Tree (SMT), where the cost of each link is the latency of the link. The operations in ALMI greatly depend on the central server, which collects the latency information for the entire session, construct a SMT, and then communicates results back to all members. Such a centralized control approach simplifies the routing problem compared to distributed approaches; however, it works well for a small group and inevitably suffers from the single-point failure problem.

Project	Objective	Structure	Approach
Scattercast	min latency	proxy-based	mesh-first
Overcast	max bandwidth	proxy-based	direct
Narada	min latency	end-to-end	mesh-first
ALMI	min system cost	end-to-end	direct

Table 2.1 Comparison of Four Overlay Multicast Projects

2.2 IP Multicast Congestion Control

IP Multicast is a technology that enables group communications in IP networks while preserving bandwidth in an efficient manner, especially for large groups. However, deployment of IP multicast on the Internet has not been as rapid as desired by the user community [19]. One of the reasons for such slow deployment is the lack of efficient multicast congestion control schemes.

A congestion control algorithm fairly distributes network resources under various load and fault conditions. Congestion control for multicast applications is an active and important area of current research. It is far more difficult to find a counterpart of TCP's congestion control mechanism compared with unicast case. With heterogeneous receivers, multicast first presents a challenge in how to economically, accurately and speedily collect feedback information from them. And since a badly designed algorithm used by multicast can cause more damage to the network during congestion, it is very important for multicast congestion control to be very robust, and to demonstrate it will share network resources fairly with other traffic.

First we present the elements of multicast congestion control architecture, and different combination of these elements constitutes different schemes. A congestion control scheme for multicast video possesses specific requirements for these elements. These requirements are discussed, along with the evaluation criteria for the performance of multicast video. Then we categorize the schemes we present into end-to-end schemes and router-supported schemes and present a number of schemes in these two categories.

2.2.1 Architecture Elements of IP Multicast Congestion Control

In this section, we present the elements that const itute multicast congestion control architecture. Detailed studies of these elements can be found in [29, 30, 31, 32].

Reliable Multicast vs. Real-Time Multicast — Reliable multicast is used by applications such as "updating software" in which accurate delivery of data is required and it allows acceptable delay or low throughput. While real-time multicast is used by applications such as video conferencing in which some packet loss can be tolerated for the sake of higher throughput and/or lower delay. The difference between these two types of multicast results in selecting different sets of the following architectural elements.

Window-Based vs. Rate-Based — Window-based schemes use a TCP-style congestion window at the sender or receiver(s) to control the load they provide to the network. And rate-based schemes use the source's rate as the regulating parameter. The rate is calculated based on either a model or using a simple increase/decrease algorithm and is kept below a level based on some congestion feedback from the network. In the model-based case, the feedback represents measurements of some parameters that are used in model calculations. In the simple increase/ decrease case, the feedback acknowledges whether there is congestion in the network or not. The most general algorithm for increase/decrease is Additive Increase Multiplicative Decrease (AIMD) [33]. Most video multicast applications are rate-based. A detailed discussion of the difference between window-based and rate-based multicast congestion control can be found in [32].

Sender-based vs. Receiver-based — Multicast congestion control schemes can be classified into two categories: sender-based (single-rate) schemes and receiver-based (layered multi-rate) schemes. In sender-based schemes, the sender performs all the flow/congestion control functionality using a single stream to all receivers that receive the same rate. While in receiver-based schemes, the source data is sent in layers and receivers are then allowed to receive as many layers as they can.

Feedback Mechanism — All reliable multicast applications must have a feedback mechanism to insure the correct delivery of congestion information. Some real-time schemes use feedback as well. Current multicast congestion schemes mainly rely on

feedback provided by the receivers. In [35, 36] the authors propose an alternative of relying on network Explicit Congestion Notification to provide feedback to the sender.

End-To-End vs. Network-Supported — End-to-end schemes rely on the collaboration among the sender and receiver(s) and no support from network in needed. While network-supported ones gain support from network in the form of agents or special router mechanisms. Comparatively end-to-end schemes offer the advantage of possible immediate deployment in the best-effort Internet.

2.2.2 Evaluation of Multicast Video

In this section, we collect the criteria commonly used to evaluate different video multicast protocols.

Dealing with Heterogeneity — This means developing a methodology that enables the sender to communicate with different receivers and satisfy their requirements simultaneously. In multicast video case, receivers may require different levels of quality of the video information, which translates into different rates delivered to them. This is called multi-rate multicast and we introduce it in detail later.

Scalability — In the case of multicast video, the number of receivers is unknown to the sender and may grow significantly. In general, feedback mechanisms in multicast applications are the main source of scalability problems.

TCP-friendliness — Fairness between competing protocols on the Internet is a serious issue and a very critical one for the robustness of the Internet. In particular, new protocols should prove to be TCP-friendliness before their deployment on the Internet. Most multicast video applications are based on UDP, which is known to be unfair to TCP because UDP simply does not have a congestion control mechanism. This implies that TCP-friendly congestion control must be provided by a protocol at a level higher than UDP.

Other Criteria — In addition to these criteria, there are other criteria that are important as well, i.e., the utilization of the available bandwidth at the receivers, stability of the quality of the received video, and time to converge to this stable level of quality at the receiver, etc.

2.2.3 End-to-End Schemes

In this section we divide end-to-end schemes into two categories: single-rate schemes and layered (multi-rate) schemes.

Single-Rate Multicast

In sender-based schemes, a single rate is sent to all receivers and the feedback is collected from receivers to change the sending rate. Different methods are proposed in the literature for feedback consolidation and control. We list here the main works in this area.

SFC (Scalable Feedback Control) [38, 39] used feedback messages from receivers with information on packet loss to estimate the "group" reception status. In DSG (Destination Set Grouping) [34] receivers are divided into sets corresponding to different streams (replicated by source, with different rates) and feedback is collected from each set so that the rate for this set is adjusted to meet its receivers' capabilities. PGMCC (Pragmatic General Multicast Congestion Control) [40] uses window-based TCP-like congestion control based on positive ACKs that are exchanged between the sender and a group of representatives called the *ackers*. An extension for equation-based congestion control to multicast applications was recently presented in [41], which is called TFMCC (TCP-Friendly Multicast Congestion Control). This approach requires calculation of the round-trip time and collecting and processing feedback.

Some work has been done on the optimization-based approach for single-rate multicast. Most proposals and studies in this area are either based on simulations or

based on experimental implementations. Limited attempts were made to model this problem analytically. In [42] congestion control for single-rate multicast is formulated as an optimization problem. Optimization-based work for the multi-rate (layered) case can be found in [5], and we present it later as part of our discussion on layered multicast. In [42] the authors adopt an economic theory for utility maximization to formulate the multicast congestion control as the problem of maximization of the aggregation of receivers' utility. The authors present a utility function that takes receivers' interests into account. While this work does not provide specific architectures or algorithms, it helps guide the development of these architectures and algorithms by linking their performance to the overall performance of the network in a formal manner.

Layered Multicast

Layered multicast is based on the ability to generate the source data in a layered format and to send the layers as different multicast groups. Each layer contains a subset of the information being sent. Receivers decide on how many layers (or equivalently, multicast groups) they can join using bandwidth inference techniques. Layers should be joined in a *cumulative* manner, which means joining them in order of their relevance. The lowest layer contains the minimum information necessary to achieve basic quality, and each subsequent layer provides progressive enhancement. So the lowest layer should be joined first.

In the following we present a brief description of some of the proposed layered multicast schemes that are most commonly cited and that represent the main methodologies for this group of schemes.

RLM (Receiver-driven Layered Multicast) [9] is one of the earliest proposals for layered multicast and it is most commonly cited. The sender sends each video layer to a separate IP multicast group and each receiver subscribes to a certain set of video layers by joining the corresponding IP multicast group. LVMR [10] is a system for

distributing MPEG-encoded video, which deals with heterogeneity in similar way as RLM and offers two major contributions to the area of layered multicast. First, it adjusts the video reception rate at receivers using a hierarchy of agents in the network that control the rate. Second, it introduces the concept of recovery using retransmission from designated local receivers to reduce recovery time. In an attempt to address the fairness issues of RLM, the authors of [45] proposed the Receiver-driven Layered Congestion (RLC) control protocol, a TCP-like congestion control scheme for multicast applications. PLM (Packet-pair Layered Multicast) [46] is based on the generation of packet pairs (PP) to infer the available bandwidth. In PLM, source packets are generated and sent in pairs and by measuring the delay between the two packets in subsequent pairs, receivers can infer network congestion status.

In [5] the authors present a formulation of the multi-rate multicast problem as an optimization problem. The objective is to achieve rates that maximize the aggregated receiver utility in multi-rate multicast sessions. The work is targeted at finding a trade-off between bandwidth utilization and fairness. The authors provided two algorithms to solve the optimization problem. Although the work in [5] is not architectural in nature, the authors provided guidelines on the implementation of these algorithms in a real network with simulation results of testing the convergence of the algorithms.

2.2.4 Router-supported Schemes

In this section, we present multicast congestion control mechanisms that rely on router support. We classify these mechanisms into two categories. The first category is sender-based, single-rate schemes that rely on *packet filtering* at the router. Filtering is dropping packets at the routers during congestion based on some criteria such as the priority of the packet. The second category is multi-rate (layered) schemes that rely on sending the data in layers and letting routers (rather than senders or receivers) control

the subscription to the layers and the flow control of each layer.

Single-Rate with Packet Filtering

This class of schemes is usually based on an active queueing mechanism which is different from the normal queueing operation of the router in that it discards packets during congestion based on some criteria as opposed to the traditional drop tail strategy. For reliable multicast, these schemes are usually combined with an FEC technique or a local retransmission and recovery mechanism. A general framework for these schemes is shown in Figure 2.2 (Note that in the figure we represent higher rates with thicker arrow)s.

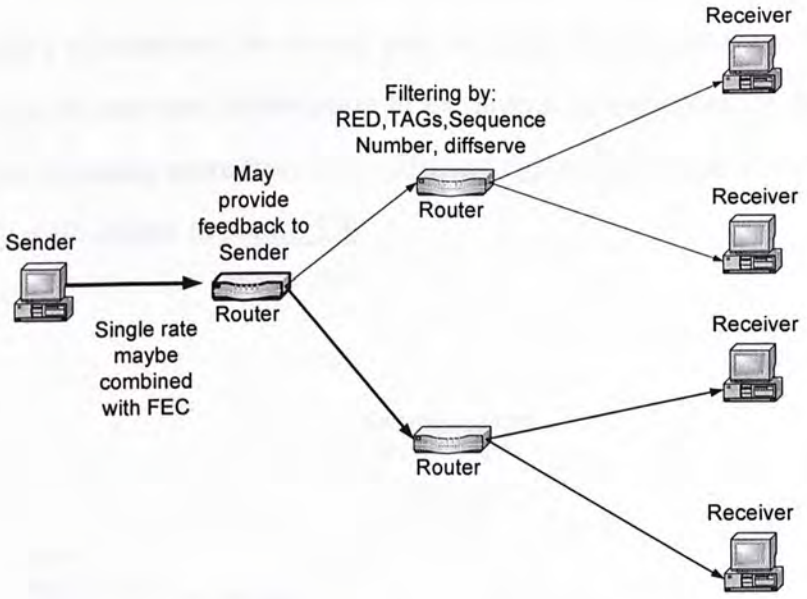


Figure 2.2 Single-rate with packet filtering

The authors in [47] present a protocol that is generally a representative of packet-filtering schemes. The filtering part of this protocol is based on a sequence number that is included in each packet. Clerget [48] presents a scheme for UDP multicast flow control that relies on a tag calculated at the source and included with the packet. Routers filter multicast flows based on this tag and drop packets that are below a certain threshold. A scheme that is based on a combination of Explicit Congestion Notification (ECN) and Random Early Detection (RED) is proposed in

[49]. The approach is called Efficient Congestion Avoidance Mechanism (ECAM). The routers detect congestion by monitoring the average queue size and act differently after comparing the queue size to RED's thresholds. A. Matrawy etc. [50] have proposed an approach for multicast congestion control that is suitable for video applications. In this approach the sender sends packets in one multicast session and marks them with different priorities. The router, based on its congestion status, will inform the sender via a feedback message about how congested the router's queue is at a certain priority level. The sender uses this information to change its rate and the ratio between packets marked with different priorities.

Multi-Rate with Router Flow/Congestion Control

In this category of schemes, the data is sent in layers. Routers manage these layers and keep track of receivers' subscription to the layers. In this category, the overhead on the routers is usually more than in the filtering approach. A general framework for this category is illustrated in Figure 2.3.

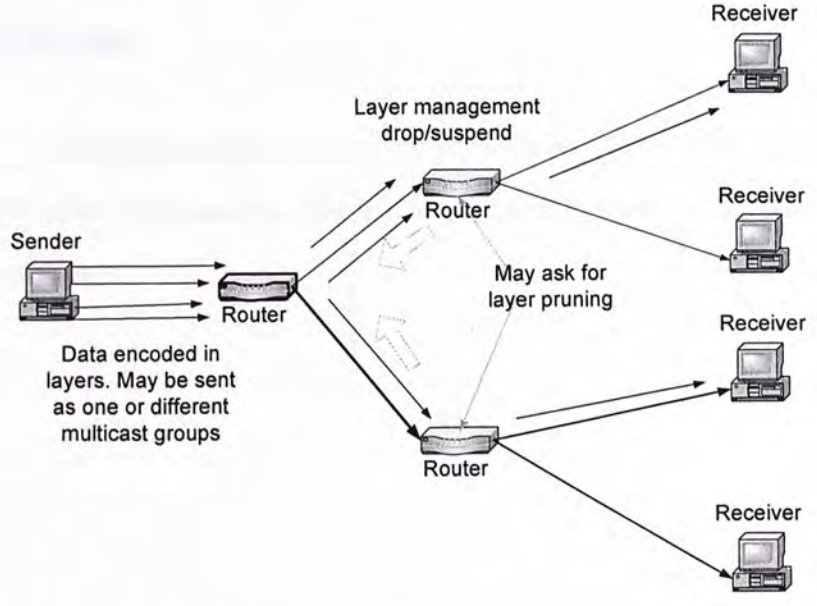


Figure 2.3 Multi-rate with router congestion control

The authors of [51] proposed Receiver-Selectable Loss Priorities (RSLP) as a means to implement a simple two-level priority-dropping scheme at the routers. During

congestion, the router attached to the congested link drops packets associated with groups mapped as higher priority at this router. In the same fashion as in [48], the authors of [52] propose a scheme called Network-supported Layered Multicast (NLM). In NLM the sender hierarchically encodes the data into several CBR layers and sends all of them in a single multicast session. The layer number is included in the header of the packet. Receivers join a session with all its layers and it is up to the router to decide on how many layers to send to the receiver based on the congestion status of the router. Router-Assisted Layered Multicast (RALM) [53] is another variant of layered multicast that is based on router assistance. The basic idea of RALM is router-controlled suspension/retry for layered multicast. An RALM-aware router monitors the queue status of each of its outgoing links. If congestion is detected on the links, the router immediately suspends some of the current transmitted groups on that link temporarily. The choice of which group is suspended is based on the importance of the data set by the sender. Routers will try to re-activate a suspended group on an outgoing link when congestion is relieved on this link.

2.2.5 Conclusion

In this section, we have presented a survey of multicast congestion control schemes especially for video applications. The tree in Figure 2.4 shows a classification of the presented schemes.

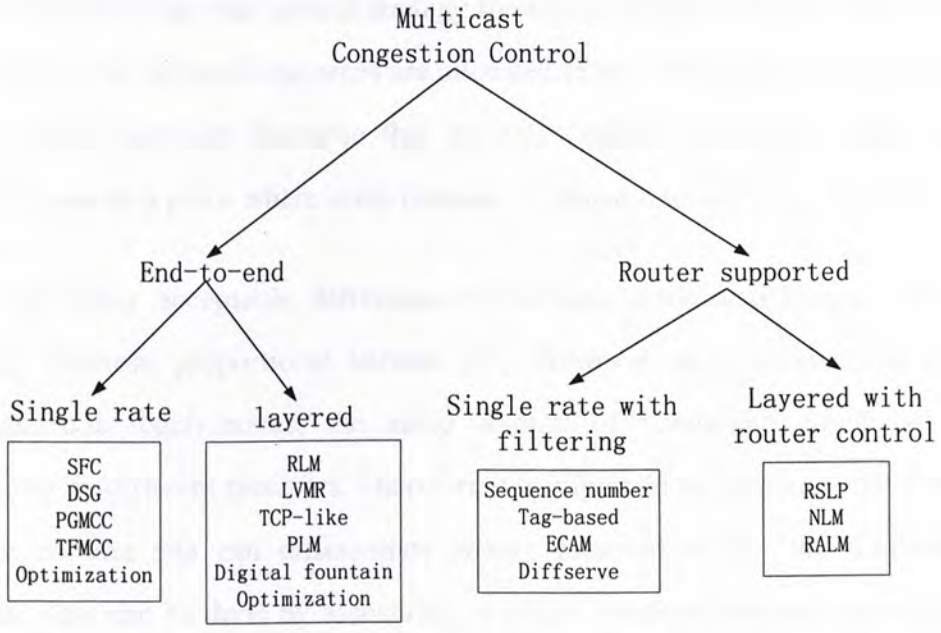


Figure 2.4 Classification of multicast congestion control schemes

2.3 Optimization-based Rate Control in IP unicast and multicast

In this section we discuss optimization-based rate control in both IP unicast and multicast, which is the base of our work.

Optimization-based rate control has been extensively investigated in the context of IP unicast and multicast. For elastic traffic sources, effective rate control is required to control congestion in a communication network. Elastic traffic sources are those which do not require a fixed rate of service and can adjust their transmission rates based on the congestion level of the network. Examples of elastic traffic sources include internet traffic sources using TCP, and sources using ABR service in ATM networks.

An effective rate control strategy should ensure that traffic offered to a network by different traffic sources remain within the limits that the network can carry. Besides

ensuring stability, the rate control strategy should ensure efficient use of the network, and also that the network resources are allocated to the competing flows in some fair manner. It is therefore desirable that the rate control algorithm would steer the network towards a point where some measure of global fairness is maximized.

There are many acceptable definitions of fairness, some well-known ones being max-min fairness, proportional fairness [43]. However, since receivers could have heterogeneous requirements, the same amount of bandwidth could be valued differently by different receivers. Therefore it is important to generalize the notions of fairness so that one can differentiate among receivers within the framework of fairness. This can be done by associating a utility function (assumed to be concave) with each receiver, which could be a measure of the perceived quality of audio/video, the user satisfaction, etc. One possible fairness objective, as advocated by Kelly in [43], is to allocate bandwidths such that they maximize the sum of the user utilities, subject to the link capacity constraints. This is also the problem that we address in both this section and our own work. The rate control algorithms people proposed achieve the optimal rates for this total user utility maximization problem. Even if all the utility functions are the same, it can be shown that various fairness objectives can be realized with the framework for different choices of the utility functions (for example. If all the utility functions are logarithmic and same for all users, the achieved rates are proportionally fair [43]).

2.3.1 Optimization-based Rate Control for Unicast Sessions

Here we present the problem in unicast case formally and describe some existing algorithms to solve this problem.

Consider a network consisting of a set $\mathcal{L}$ of unidirectional links, where link $l \in \mathcal{L}$ has capacity c_l. The network is shared by a set $\mathcal{S}$ of unicast sessions (users). Let $\mathcal{L}_s \in \mathcal{L}$ denote the set of links used by session $s \in \mathcal{S}$. Also let $\mathcal{S}_l \in \mathcal{S}$ denote the set

of sessions that use link $l \in \mathcal{L}$. Session s has a minimum required transmission rate $b_s \geq 0$, and a maximum required transmission rate $B_s < \infty$. Moreover, session s is associated with a utility function $U_s : \mathcal{R}_+ \to \mathcal{R}$, which is assumed to be concave, continuous, bounded and increasing in the interval $X_s = [b_s, B_s]$. Thus session s has a utility $U_s(x_s)$ when it is transmitting at a rate x_s, where $x_s \in X_s$. The objective is to maximize the "social welfare", i.e., sum of the utilities over all the sessions, subject to the link capacity constraints. The problem can be posed as:

$$\max \sum_{s \in \mathcal{S}} U_s(x_s)$$

subject to: $$\sum_{s \in \mathcal{S}_l} x_s \leq c_l \qquad \forall l \in \mathcal{L}$$

$$x_s \in X_s \qquad \forall s \in \mathcal{S}$$

In [7], Low etc. propose an algorithm based on dual approach to solve the above problem. In this algorithm, based on the aggregate rate of traffic on the link, either communicated by the sources or measured, each link in the network calculated a "link price" of its own. Then the network (links) conveys to each user all the "link prices" of the links on its path and adding them together produces its "session price". Then the user computes a rate to maximize its own profit based on this "session price". A problem with the implementation of this algorithm is that the link prices (which are basically the dual variables) are real numbers and could vary over a wide range. This poses difficulty in communicating the price to end hosts using a small number of bits in packet header.

In [26], Kar etc. propose an algorithm based on the primal approach. In this algorithm, the network communicates to the user the number of congested links on the user's path. On congestion, the user decreases its rate based on this network feedback;

otherwise it increases its rate based on the derivative of its utility function. An attractive feature of this algorithm is the simplicity of both the user and network algorithms. Moreover, the congestion information transmitted by network to the user can be conveyed in only $\lfloor \log_2 \tilde{L} \rfloor + 1$ bits, where $\tilde{L}$ is the maximum number of links of user's path. This implies that just one byte in the packet header is sufficient to carry the network congestion feedback in most networks.

In [44], the authors suggest a randomized marking based implementation of the algorithm in [7], which uses only one bit to convey the congestion feedback. Here the single congestion indication bit is marked probabilistically and independently at each link on the user's path. The user then can estimate the "session price" by seeing the proportion of marked packets. However, the authors do not provide any proof of convergence. Moreover, the randomized marking policy of [44] can be applied to [26] too, and thus the overhead in packet header can be lessened. Initial simulations in [26] indicate that the primal-based algorithm also performs well with this modification.

In [37], the authors introduce both primal and dual algorithms for this system utility maximization problem. However, these algorithms solve only an approximate version of the original problem rather than the actual problem. The authors do suggest a choice of price functions for which the solution provided by their algorithms can be made arbitrarily close to the actual solution. But this choice of price functions could make the link prices vary over a wide range and pose similar difficulties in practice as [7].

Another related, but different approach is proposed in [21]. In this work, the authors propose an additive increase-multiplicative decrease scheme for reaching the socially optimal solution. Here the user adjusts its rate based on the proportion of marked packets or end-to-end (measured) losses. However, the algorithm is presented for some specific utility functions, and it is not clear how to address the case of more

general utility functions. Also, the convergence has been proved under certain simplifying assumptions, some of which are not likely to hold in practice.

2.3.2 Optimization-based Rate Control for Multi-rate Multicast Sessions

Similarly, in this section we present a mathematical formulation of the problem in multi-rate multicast case and some existing algorithms to solve it.

Consider a network consisting of a set $\mathcal{L}$ of unidirectional links, where link $l \in \mathcal{L}$ has capacity c_l. The network is shared by a set $\mathcal{M}$ of multicast sessions. Multicast session $m \in \mathcal{M}$ is associated with a unique source, a set of receivers and a set of links specified by $\{s_m, R_m, L_m\}$. Let $\mathcal{R}$ be the set of all receivers over all multicast sessions. Also let $\mathcal{S}_l$ denote the set of receivers that use link $l \in \mathcal{L}$. Receiver $r \in \mathcal{R}$ is associated with a utility function $U_r(x_r)$, where x_r is the rate at which r receives data. Let $b_r \geq 0$ and $B_r < \infty$ be the minimum and maximum rates, respectively, required by receiver r. Let $X_r = [b_r, B_r]$ denote the interval in which the receiver rate x_r must lie. The objective is to maximize the "social welfare", subject to the link capacity constraints. The problem can be posed as:

$$\max \sum_{r \in \mathcal{R}} U_r(x_r)$$

$$\text{subject to:} \quad \sum_{m \in \mathcal{M}} \max_{r \in \mathcal{S}_l \cap R_m} x_r \leq c_l \qquad \forall l \in \mathcal{L}$$

$$x_r \in X_r \qquad \forall r \in \mathcal{R}$$

Very recently, the problem of fair allocation of resources in multi-rate multicast networks has received considerable attention. However, most of the research in this context is concerned only with the notion of max-min fairness (see [23], [24], [25]). Among these, [24] considers the problem of utility allocation. Whereas our objective is maximizing the aggregate utility, the one in [24] is to allocate the utilities fairly.

Although utility maximization has been extensively studied within the context of unicast rate allocation to achieve congestion control [7, 26, 37, 44], relatively fewer studies approached the multicast rate allocation problem via solving a general utility maximization formulation, with the notable examples being [5,6].

It is worth noting that certain factors make the multi-rate multicast problem significantly different and more complex than its unicast version. In the unicast version of this problem, the link constraints are linear and the problem is separable, which is amenable to distributed solutions. While in multicast version, the problem contains some max functions. The max functions, besides being nonlinear, couple several variables together, making the problem non-separable. Moreover, the max functions are non-differentiable. All these factors make the problem significantly different from its unicast version. Obtaining a distributed and scalable solution becomes a challenging task.

In [5], the authors propose a dual-based solution approach like the one used in [7] for the unicast case. In fact, in the special case where all the sessions are unicast, the two algorithms they propose reduce to the algorithm proposed in [7]. In [5], the authors note that the max functions are piecewise linear, and hence the constraint set can be replaced by a set of linear inequalities. So they propose an alternative formulation of the original problem by linearizing the constraint set and thus make the problem separable, which makes a distributed solution become possible. Moreover, in the solution approach in [5], a source/junction/receiver node only needs to communicate with its parent or children nodes. Thus the solution is scalable, and conforms with the existing standard.

In [6], the authors propose a primal-based approach like the one used in [26] for the unicast case. The algorithm in [6] is developed using non-differentiable optimization methods, particularly those based on subgradients. The motivation, derivation and analysis of this algorithm draw from results in subgradient optimization theory. The problem of non-separability (as well as non-differentiability) of the constraint functions can be effectively handled using subgradients. So using subgradients, the authors develop a simple distributed solution to the non-separable problem of multicast case. Comparing with the dual-based algorithm ([5]), in the primal-based algorithm proposed in [6], both the user and the network sub-algorithms are extremely simple and the overhead of communication between the network and the user is very low. What's more, the primal-based algorithm does not need to maintain per-session states at the network links.

In [22], the authors discover properties of optimal solutions of this problem. Based on these properties they describe a market-based approach that achieves an optimal solution through a decentralized convergent iterative procedure. The market-based approach differs distinctly from the approaches in [5] and [6] because it adopts a hierarchical architecture as opposed to the flat architecture of [5] and [6], and because it employs a Tatonment process different from that of [5] and [6]. The key features of this approach includes: 1) the determination of optimal link price sharing by the network users; 2) the description of a market-based mechanism which achieves a welfare maximizing solution based on the properties of optimal link price sharing hierarchical architecture.

Chapter 3 Overlay Multicast Rate Control Algorithms

3.1 Motivations

In the past decade, IP multicast has been proposed to implement multicast in an efficient way. Although IP multicast has conceptual simplicity and various benefits, its deployment is not as fast as people expected. *Overlay Multicast* has become a hot topic recently and it is a good alternative for IP multicast for its feasibility and flexibility.

We have briefly reviewed the rate control problem in IP multicast in Chapter 3. Our work is inspired by the previous work called *layered multicast streaming* [9] which is designed to implement multi-rate multicast in network layer. A stream is encoded into multiple layers and different users use different set of layers to recover the stream. Fair resource allocation in such a system is indeed accomplished by certain rate control mechanisms. One disadvantage is that users can only select from a discrete set of rate values, which brings coarse granularity to resultant rates. While in *Overlay Multicast*, data relay occurs at end hosts, which have capabilities far beyond basic ones like storing and forwarding. So besides the layered approach, the end host can be programmed to implement various kinds of end-to-end streaming adaptation techniques. Therefore users can select from a continuous range of rate values which provides finer granularity than the layered case before. Thus we are motivated to

investigate the optimal resource allocation problem in *Overlay Multicast*. Due to the intrinsic difference from IP multicast, we are faced with some new challenges. Details are described in subsequent sections.

3.2 Problem Statement

3.2.1 Network Model

In this section we describe the network model and formulate the optimal resource allocation problem as a convex programming problem.

Consider an overlay network formed by self-organized end hosts. It is shared by a set of M multicast groups (sessions). Session m has a unique source host S_m, a set of receiver hosts R_m, and a set of overlay links L_m that forms the overlay multicast tree. Thus any multicast session can be represented by $\{S_m, R_m, L_m\}$. Note that in our work, we do not address the choice of algorithm in the tree construction. Instead, our work is based on a ready-made tree no matter how it is built. Moreover, each multicast session independently executes resource allocation on its participating parties with necessary per-session state information stored at end hosts. So investigating one session's case is general enough to give us insight into the problem of concern.

From the framework of Overlay Multicast in the last chapter, we know that an overlay multicast session consists of a set of unicast flows $\mathcal{F}=\{1,2,...,F\}$ with a corresponding set of receivers $\mathcal{R}=\{1,2,...,R\}$. The number of unicast flows is equal to the number of receivers in the tree, i.e., $F=R$. Each receiver $r \in \mathcal{R}$ will specify its minimum and maximum required rate based on its own requirement and capability.

Correspondingly each flow $f \in \mathcal{F}$ has a rate x_f which lies in an interval $X_f = [b_f, B_f]$, where $b_f \geq 0$, $B_f < \infty$, $b_f, B_f \in R$. Also, flow f is associated with a utility function $U_f(x_f)$, which is assumed to be strictly concave, bounded and continuously differentiable in the interval X_f.

Each unicast flow corresponds to an overlay link which consists of a set of underlying network links (note that below we will use "network link" and "link" interchangeably). For flow f, we use $L(f)$ to denote its constituent links. Assume that the multicast session totally has L network links, denoted by $\mathcal{L} = \{1, 2, ..., L\}$. Each link $l \in \mathcal{L}$ has a bandwidth capacity c_l. Correspondingly, for link l, we use $F(l)$ to denote the set of flows passing through it.

3.2.2 Problem Formulation

Our objective is to maximize the "social welfare", i.e., the aggregation of utilities over all flows, subject to some constraints brought about by the inherent nature of the overlay multicast scheme.

One constraint is caused by the finite capacity of network link. The aggregation of rates over all flows in $F(l)$ cannot exceed c_l. This is also a common constraint in IP resource allocation problems [5, 6, 7, 8].

Another constraint is due to the special relay role of end hosts in overlay multicast. Aside from those located as leaves in the tree, all receivers also act as senders for their downstream receivers. Thus a unique challenge in overlay multicast is that the bandwidth and data availability (data rate) of each receiver are constrained and heterogeneous, which further limits the data rates of its downstream receivers when it acts as the supplying peer. In other words, the rate of a flow cannot exceed the rate of

its father flow. In our work, for each flow f, we use π_f to denote its father flow and $C_f = \{f' : \pi_{f'} = f\}$ to denote the set of its child flows in the tree.

Bearing these in mind, it is reasonable to allocate resources (link capacities) in an overlay multicast session to solve the following optimization problem:

Maximize: $\sum_{f \in F} U_f(x_f)$

Subject to: $\sum_{f \in F(l)} x_f \le c_l \qquad \forall l \in \mathcal{L}$ (i)

$x_f \le x_{\pi_f} \qquad \forall f \in \mathcal{F}$ (ii)

$x_f \in [b_f, B_f] \qquad \forall f \in \mathcal{F}$ (iii)

To put the problem in a more standard format for nonlinear programming theory, we define two new matrices corresponding to constraints (i) and (ii). For (i), we define a binary matrix A of size $L \times F$, where $a_{ij} = 1$ if flow j flows through link i; otherwise $a_{ij} = 0$. For constraint (ii), we define a $F \times F$ matrix B, where

$$b_{ij} = \begin{cases} 1 & i = j \,\&\, \pi_i \neq \phi \\ -1 & j = \pi_i \\ 0 & otherwise \end{cases}.$$

B is used to express the data relay relationship in the tree structure. Moreover, we collect all the c_l into a vector $\vec{c}$, collect all the x_f into $\vec{x}$ and collect all the X_f into $\vec{X}$. Having these new definitions, we can write our problem in a more formal way:

Problem **P**:

Maximize: $\sum_{f \in F} U_f(x_f)$

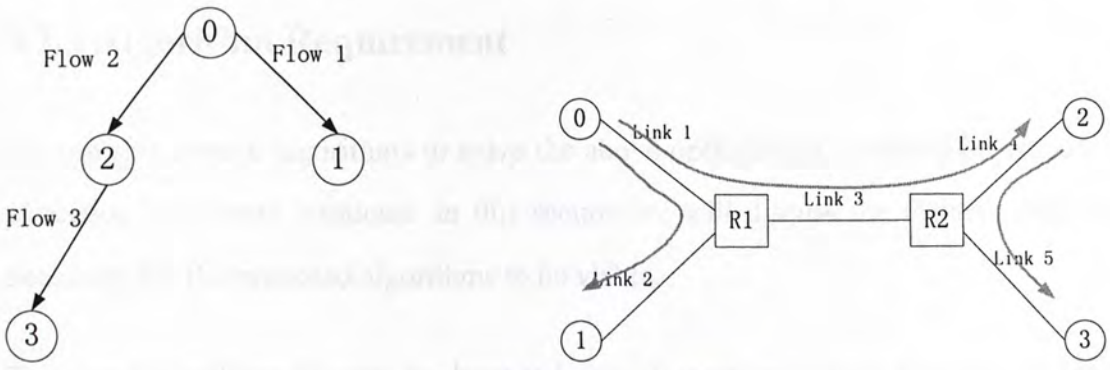
Subject to: $A\vec{x}' \leq \vec{c}$ (1)

$B\vec{x}' \leq 0$ (2)

$\vec{x} \in \vec{X}$ (3)

Note that in constraint (3), $\vec{X}$ is the aggregation of a set of bounded continuous intervals in multi-dim space, which means x_f can be of any real value in X_f. In conventional IP multi-rate multicast using layered approach, the resultant rate can only be selected from a discrete set of values [5, 6, 7, 8]. But constraining X_f to a finite number of discrete points will destroy the convexity of the problem (it becomes an integer programming problem), which is crucial for developing a distributed solution [20]. So people still use the above "convexified" formulation to compute rates and then round these rates to the allowed discrete levels. Such a process definitely reduces the precision of resultant rates. While in overlay multicast case, as we said, the receiver rates can be continuous. So we can obtain a precise result by solving the above problem.

Now we use a simple example shown in Figure 3.1 to illustrate the problem formulation.



(a) Overlay Multicast Tree (b) Corresponding Network Topology

Fig 3.1 Overlay Multicast Example

The overlay multicast tree structure is illustrated in (a). The tree consists of three unicast flows (F=3) and five physical links (L=5). Each flow consists of several links as illustrated using red lines in (b). The capacity of each link is marked beside the link. And each flow's rate ranges from 1(unit of bandwidth) to the bandwidth of the link attached directly to the corresponding receiver. Therefore the optimization problem for this multicast session is:

Maximize: $\sum_{i=1}^{3} U_i(x_i)$

$$\begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \le \begin{bmatrix} 8 \\ 6 \\ 15 \\ 10 \\ 4 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & -1 \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \le \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Subject to: $1 \le x_1 \le 6$

$1 \le x_2 \le 10$

$1 \le x_3 \le 4$

3.2.3 Algorithm Requirement

We use rate control algorithms to solve the above optimization problem for resource allocation in overlay multicast. In this section we will discuss the features that are necessary for the proposed algorithms to be viable.

To solve the problem directly, we have to know all receivers' utility functions and the complete topology information of both application-layer and network-layer. In a large network such as the Internet, such information is not available centrally. Thus, it is important to devise distributed solutions, where each receiver adapts its rate based only on local information and reaches the system optimum without any centralized coordinator.

Another concern is the issue of scalability. A solution would not scale if, for example, a node in the multicast tree has to maintain some state information for all downstream nodes of the tree. Since the size of the tree can be very large, this might lead to tremendous processing/storage pressure on such a node, particularly if the node is close to the source. Therefore we would like to have a solution where the processing/storage overhead at a node (end host) in an overlay multicast tree does not depend significantly on the size of the session.

Comparing to IP multicast, one of the advantages of overlay multicast is that it avoids any change to the existing infrastructure by transferring all functions to end hosts [18]. So basically our algorithm should obey such a rule and will not induce any extra changes.

It is also desirable that the overhead of information exchanged (required for the optimization process) between end hosts is as low as possible, so that it only occupies a few bytes in the packet header. We should also reduce other kinds of overhead (discussed below) to facilitate the algorithm's deployment in a real network environment without appending much burden.

With the assumption that utility functions are strictly concave, P becomes a typical convex optimization problem with linear constraints. From classical optimization theory, such problems admit a unique solution under such an assumption [20]. Two popular algorithms arise from previous work: primal-based algorithm [6] and dual-based algorithm [5]. Based on them, below we present two kinds of algorithms for solving P.

3.3 Primal-based Algorithm

Now we present a distributed primal-based algorithm to solve the optimization problem P described above. The basic idea is taken from [27], where an iterative subgradient-based optimization method has been proposed for a general class of convex optimization problems. However, the optimization procedure in [27], if implemented in our case, would require centralized coordination, and therefore violate the algorithm requirement described in the last section. Our algorithm presented below is a modified version of the algorithm in [27] which is amenable to a distributed implementation and yet retains the convergence properties of the original algorithm. We'll also give a convergence analysis later in this section.

3.3.1 Notations

In this part we define some notations that will be used in this algorithm.

A link l is said to be congested when the total traffic assigned to it exceeds its capacity c_l. We define an indicator variable for each link to denote whether the link is congested or not, i.e.,

$$\varepsilon_l = \begin{cases} 0 & if \quad \sum_{f \in F(l)} x_f \le c_l \\ 1 & if \quad \sum_{f \in F(l)} x_f > c_l \end{cases} \tag{4}$$

We will refer to the variable ε_l as the "link congestion indicator" for link l.

For each flow $f \in \mathcal{F}$ we define a variable e_f to indicate the number of congested links passed by flow f. So $e_f = \sum_{l \in L(f)} \varepsilon_l$.

Now we interpret ε_l as the penalty to be paid by each flow using link l for congesting l. Then e_f is the total penalty to be paid by f for congesting the links in $L(f)$. Let this penalty be charged to f's associated receiver $R(f)$. As we said before, except those located as leaves, each receiver helps its child receivers in relaying traffic for them, so the penalty charged to $R(f)$ should be split among itself and its children receivers. Thus for each receiver r, combined with its child receivers, we get a set of receivers to investigate how to split penalty among them. For the algorithm to work correctly, the splitting factors α_r for all r in such a set should satisfy the following conditions: $\alpha_r \geq 0$; $\sum_r \alpha_r = 1$. Also, the splitting factor of a receiver is zero if its receiver rate is not the same as its parent receiver in the set. Since the rate of the parent receiver is the maximum in this set, this implies that the penalty at the parent receiver is split amongst only those receivers who are receiving the maximum rates. Two integers (s and t) are enough to tell a receiver its shared penalty (s/t) for its parent receiver. We will show the calculation of s and t in our algorithm description in Figure 3.2.

Then we can define a variable P_f for each flow $f \in \mathcal{F}$. P_f is the final penalty charged to $R(f)$ after calculations illustrated in Figure 2. We will refer to P_f as "flow congestion penalty".

Aside from the congestion penalty, for each flow, we define another penalty R_f which is defined as:

$$R_f = \begin{cases} 1 & x_f > x_{\pi_f} \\ 0 & x_f \le x_{\pi_f} \end{cases} \tag{5}$$

We will refer to R_f as the "data availability penalty".

3.3.2 An Iterative Algorithm

The algorithm we propose here is an iterative optimization algorithm. Before we describe the algorithm, consider two positive sequences $\{\alpha_n\}$ and $\{\beta_n\}$ with the following properties:

$$\lim_{n\to\infty} \alpha_n = 0 \qquad \sum_{n=1}^{\infty} \alpha_n = \infty$$
$$\lim_{n\to\infty} \beta_n = 0 \qquad \sum_{n=1}^{\infty} \beta_n = \infty \tag{6}$$
$$\lim_{n\to\infty} \frac{\alpha_n}{\beta_n} = 0$$

Now look at an iterative procedure to solve problem P, where $x_f^{(n)}$, the receiver rate of flow f at the nth step, is updated as below.

$$x_f^{(n+1)} = \left[x_f^{(n)} + \alpha_n U_f'(x_f^{(n)}) - \beta_n (P_f^{(n)} + R_f^{(n)}) \right]_{X_f} \tag{7}$$

Here $[.]_{X_f}$ denotes a projection onto the set X_f.

Also note that the rate update procedure described above inherently assumes that the function U_f is differentiable in X_f. This, in general, is not necessary. If $U_f'(x_f)$ does not exist at some point $x_f \in X_f$, it can be replaced by a subgradient of U_f at x_f.

The iterative rate update procedure, stated above, has a simple intuitive interpretation.

In the procedure, the receiver rate of a flow is increased according to the derivative of the flow's utility function, while it is decreased according to the flow's "flow congestion penalty" and "data availability penalty". That's to say, when any of the links on its path or its father flow's path is congested, or its rate exceeds its father flow's rate, the flow backs off by decreasing its rate.

As we will show in next section, the step-sizes α_n and β_n need to satisfy (6) for the algorithm to have guaranteed convergence. Note that (6) roughly implies that the increment of the rate of a flow needs to be (asymptotically) much smaller as compared to the decrement.

We generalize our algorithm for links and flows in Figure 3.2. Here links and flows act like processors in a distributed computation system.

3.3.3 Convergence Analysis

Now we investigate the convergence properties of the algorithm outlined in Figure 3.2.

3.3.3.1 Assumptions

In the convergence analysis, we make similar assumptions as in [26] on the problem P.

Assumption 1: (Feasibility) The problem P is feasible. i.e. $\sum_{f \in F(l)} b_f \le c_l$ for all $l \in \mathcal{L}$ and $b_f \le B_{\pi_f}$ for all $f, \pi_f \in F(l)$. Note that in a special case when $b_f = 0$ for all $f \in \mathcal{F}$, the feasibility assumption is satisfied. Thus an optimal solution to P exists, although it may not be unique.

Link l's algorithm:

Receive x_f from all flows in $F(l)$;

Update congestion indication ε_l: $\varepsilon_l = \begin{cases} 0 & if \quad \sum_{f \in F(l)} x_f \le c_l \\ 1 & if \quad \sum_{f \in F(l)} x_f > c_l \end{cases}$;

Send ε_l back to all flows it received rates from.

Flow f's algorithm:

Receive ε_l from all links in $L(f)$;

Receive rates from father flow and children flows;

Receive assigned part of P_{π_f} denoted in s', t': $P_f' = s'/t'$;

Update penalty P_f and R_f:

$s = s' + t' \sum_{l \in L(f)} \varepsilon_l$;

Compare its rate with children's to decide: $t = t'$[(number of children flow whose rate is equal with it) +1];

Update $P_f = s/t$;

Send s, t's value to those children flow counted in t;

Update $R_f = \begin{cases} 1 & x_f > x_{\pi_f} \\ 0 & x_f \le x_{\pi_f} \end{cases}$;

Update receiver rate $x_f^{(n+1)} = \left[x_f^{(n)} + \alpha_n U_f'(x_f^{(n)}) - \beta_n (P_f^{(n)} + R_f^{(n)}) \right]_{X_f}$;

Send $x_f^{(n+1)}$ to all links in $L(f)$;

Send $x_f^{(n+1)}$ to its father flow and children flows.

Fig 3.2 link and flow algorithms (primal-based)

Assumption 2: (Bounded Slope) For every $f \in \mathcal{F}$, $a_f \le U_f'(x_f) \le A_f \quad \forall x_f \in X_f$ where $a_f \ge 0$ and $A_f < \infty$.

If U_f is non-differentiable in X_f (i.e., U_f' does not exist at all points in X_f), we will assume that Assumption 2 holds for all subgradients of U_f in X_f.

Now we state some convergence results under various conditions of the step-sizes. Also, we give complementary explanations for why these results still hold in our algorithm.

3.3.3.2 Convergence with various step-sizes

Let X^* denote the set of optimal solutions of P. Let $U(x) = \sum_{f \in \mathcal{F}} U_f(x_f)$ be the overall utility and U^* be the corresponding optimal value. Thus $U^* = U(x^*)$ for any $x^* \in X^*$. For the iterative procedure (7) and with the step-sizes satisfying (6), **Theorem 1** in [26] states that $\lim_{n \to \infty} \rho(x^{(n)}, X^*) = 0$, where $\rho(x, Y) = \min_{y \in Y} \|x - y\|$ denote the Euclidean distance of a point x from any compact set Y.

If the step-sizes are constant, we cannot guarantee convergence to the optimum in the sense stated in **Theorem 1**. However, it is possible to show a slightly weaker result, as stated in **Theorem 2** in [26]. It says that with constant step-sizes the resultant rates can converge to the neighborhood of the optimal rates.

3.3.3.3 Theorem Explanations

The proof for the above two theorems are given in [26]. Our work has a similar form and reasoning as theirs except for some different constraints specific to overlay multicast. We will investigate these constraints and show that the above theorems still

hold in our case.

One is the "congestion penalty" P_f. The difference lies in how to divide a flow's (node's) P_f (ε_r) amongst its child flows (nodes). In previous work [26], for implementation simplicity, a node r just transmits its ε_r to one of the child nodes who have similar rates with r. The authors also claim that how ε_r is divided has no effect on the convergence nature of the algorithm [26]. In our work, P_f has the same role as ε_r and we divide P_f evenly in those flows involved, so this does not ruin the convergence properties proved before.

Another is the "data availability penalty" R_f we introduce to our algorithm corresponding to constraint (2) in problem P. In such a primal-based algorithm, this is the best and simplest way we can find to satisfy such a constraint. What we do is to gradually tighten the constraint, i.e., at each step, if $R_f = 1$, flow f will decrease its rate by the amount of step-size. With very small or diminishing step-sizes such enforcement will not distort resultant receiver rates a lot. Otherwise, if we remove the R_f part from the algorithm and enforce the data rate constraint to resultant rates, there will be great deviation from the optimal solution. (We will verify this in a Chapter 4.)

In summary, the new elements brought by overlay multicast do not influence the original algorithm's convergence properties. This ensures that the above theorems still apply to our algorithm.

3.4 Dual-based Algorithm

Now we present a dual-based algorithm to solve the optimization problem **P**. Since problem **P** is separable, dual methods provide attractive approaches for obtaining

distributed solutions (see Chp. 6 of [20]). Our algorithm below is mainly based on the work in [5] concerning optimal resource allocation in IP multicast.

3.4.1 The Dual Problem

Let p_l be the dual variable ("price") associated with link constraint (2) for link $l \in \mathcal{L}$. We call p_l the *link price*. Let q_f be the dual variable ("price") associated with data rate constraint (3) for flow $f \in \mathcal{F}$. We call q_f the *relay price*. Also let π_f denote flow f's father flow and C_f denote f's children flow set. Let $\vec{p} = (p_l, l \in \mathcal{L})$ and $\vec{q} = (q_f, f \in \mathcal{F})$ be the price vectors.

The lagrangian for problem **P** is

$$
\begin{aligned}
& L(\vec{x}, \vec{p}, \vec{q}) \\
&= \sum_{f \in \mathcal{F}} U_f(x_f) - \sum_{l \in \mathcal{L}} p_l \Big(\sum_{f \in \mathcal{F}} a_{lf} x_f - c_l \Big) - \sum_{f' \in \mathcal{F}} q_f \,' \Big(\sum_{f \in \mathcal{F}} b_{f'f} x_f - 0 \Big) \\
&= \sum_{f \in \mathcal{F}} U_f(x_f) - \sum_{f \in \mathcal{F}} x_f \sum_{l \in \mathcal{L}} p_l a_{lf} - \sum_{f \in \mathcal{F}} x_f \sum_{f' \in \mathcal{F}} q_{f'} b_{f'f} + \sum_{l \in \mathcal{L}} p_l c_l
\end{aligned}
\tag{8}
$$

Note that we have defined $\vec{X}$ to be the aggregation of a set of bounded continuous intervals in multi-dim space, i.e. $\vec{x} \in \vec{X}$. The dual of the problem P is:

$$
P' : \qquad \min_{\vec{p}, \vec{q} > 0} D(\vec{p}, \vec{q}) \tag{9}
$$

where the dual objective function $D(\vec{p}, \vec{q})$ is given as :

$$
D(\vec{p}, \vec{q}) = \max_{\vec{x} \in \vec{X}} L(\vec{x}, \vec{p}, \vec{q}) \tag{10}
$$

Thus we can calculate each flow's rate x_f from letting $\frac{\partial L(\vec{x},\vec{p},\vec{q})}{\partial x_f}=0$, i.e.

$$\frac{\partial L(\vec{x},\vec{p},\vec{q})}{\partial x_f}=U_f^{'}(x_f)-\sum_{l\in\mathcal{L}}p_l a_{lf}-\sum_{f'\in\mathcal{F}}q_{f'}b_{f'f}=0$$

According to the definition of matrix A and B, $a_{lf}=1$ if flow f flows through link l, $b_{f'f}=1$ when $f'=f\,\&\,\pi_f\neq\phi$ and $b_{f'f}=-1$ when $f=\pi_{f'}$. Thus we can simplify the above expression

$$\frac{\partial L(\vec{x},\vec{p},\vec{q})}{\partial x_f}=U_f^{'}(x_f)-\sum_{l\in L(f)}p_l-(q_f-\sum_{f'\in C_f}q_{f'})=0$$

For flow f, we define $P_f=\sum_{l\in L(f)}p_l$ as *flow price* and $Q_f=q_f-\sum_{f'\in C_f}q_{f'}$ as *data price*. Then we can derive the expression as followed:

$$\frac{\partial L(\vec{x},\vec{p},\vec{q})}{\partial x_f}=U_f^{'}(x_f)-P_f-Q_f=0$$

$$x_f(\vec{p},\vec{q})=[U_f^{'(-1)}(P_f+Q_f)]_{X_f} \tag{11}$$

Substituting (11) into (10) and (8), we have

$$D(\vec{p},\vec{q})=\sum_{f\in\mathcal{F}}U_f(x_f(\vec{p},\vec{q}))-\sum_{f\in\mathcal{F}}x_f(\vec{p},\vec{q})*(P_f+Q_f)+\sum_{l\in\mathcal{L}}p_l c_l \tag{12}$$

As these expressions show, the problem of finding $\vec{x}$ to maximize the Lagrangian $L(\vec{x},\vec{p},\vec{q})$ can be decomposed into separate flow optimization problems for each flow $f\in\mathcal{F}$. Such decomposition is possible due to the separable nature of the problem P. Note that in order to calculate its receiver rate x_f, flow f only needs to know the *link price* of the links it flows through and the *relay price* of its child flows. Thus

such a solution approach is distributed in nature.

3.4.2 Subgradient Algorithm

The subgradient algorithm we propose here is based on the subgradient method developed by N. Z. Shor, among others.(see [28] (Chap. 2) for a detailed discussion on this method). In our problem, since we assume that U_f is strictly concave, $D(\vec{p},\vec{q})$ is continuously differentiable and its gradient exists. Without loss of generality, we use the subgradient algorithm here in case that $D(\vec{p},\vec{q})$ is not differentiable. So the components of the gradient ∇D at $(\vec{p},\vec{q})$ are obtained as

$$\nabla D\Big|_{p_l} = c_l - \sum_{f\in F(l)} x_f(\vec{p},\vec{q})$$

$$\nabla D\Big|_{q_f} = x_{\pi_f}(\vec{p},\vec{q}) - x_f(\vec{p},\vec{q})$$

where $\nabla D\Big|_{p_l}$ and $\nabla D\Big|_{q_f}$ are the components of ∇D for p_l and q_f respectively, and $x_f(\vec{p},\vec{q})$ are such that they attain the maximum in Equation (12). Now applying the gradient projection method with a constant step-size α, the price update procedures at the n th step becomes

$$p_l^{(n+1)} = [p_l^{(n)} + \alpha(\sum_{f\in F(l)} x_f(\vec{p},\vec{q}) - c_l)]_+ \tag{13}$$

$$q_f^{(n+1)} = [q_f^{(n)} + \alpha(x_f(\vec{p},\vec{q}) - x_{\pi_f}(\vec{p},\vec{q}))]_+ \tag{14}$$

Combining (10),(13),(14), the dual-based algorithm for solving P is completely illustrated in Figure 3.3. From this we discover some new practical meanings contained in this algorithm.

Link l's algorithm:

Receive x_f from all flows in $F(l)$;

Update *link price* $p_l^{(n+1)} = [p_l^{(n)} + \alpha(\sum_{f \in F(l)} x_f(\vec{p},\vec{q}) - c_l)]_+$;

Send $p_l^{(n+1)}$ back to all flows it received rates from.

Flow f's algorithm:

Receive $p_l^{(n+1)}$ from all links in $L(f)$;

Receive rates from its father flow;

Receive rates from children flows $f' \in C_f$;

Update *flow price* $P_f^{(n+1)} = \sum_{l \in L(f)} p_l^{(n+1)}$;

Update *relay price* $q_f^{(n+1)} = [q_f^{(n)} + \alpha(x_f(\vec{p},\vec{q}) - x_{\pi_f}(\vec{p},\vec{q}))]_+$;

For each flow $f' \in C_f$

Compute relay price $q_{f'}^{(n+1)} = [q_{f'}^{(n)} + \alpha(x_{f'}(\vec{p},\vec{q}) - x_f(\vec{p},\vec{q}))]_+$

Update *data price* $Q_f^{(n+1)} = q_f^{(n+1)} - \sum_{f' \in C_f} q_{f'}^{(n+1)}$;

Update receiver rate $x_f^{(n+1)} = [U_f'^{(-1)}(P_f^{(n+1)} + Q_f^{(n+1)})]_{X_f}$;

Figure 3.3 link and flow algorithms (dual-based)

3.4.3 Interpretation of the Prices

Up to now, we have defined four prices in dual-based algorithm: p_l (*link price*), q_f (*relay price*), P_f (*flow price*), Q_f (*data price*).

The interpretation of p_l is straightforward. As noted in [5], p_l can be interpreted as the "congestion price" of link l. Note that at optimality, from Kuhn-Tucker

conditions, $p_l > 0$ if and only if $\sum_{f \in F(l)} x_f = c_l$. Therefore, at optimality, the price of an uncongested link is zero.

Now we interpret q_f. It is called *relay price* because we can interpret it as the price a flow pays to its father flow for relaying traffic to it. From the father flow's view, this price can be deemed as its reward for helping relaying traffic for its children flows.

P_f and Q_f are both defined for flow f.

$P_f = \sum_{l \in L(f)} p_l$, which means P_f of flow f is equal to the sum of *link price* over links in $L(f)$. Intuitively P_f can be interpreted as the "congestion price" for a flow f, i.e., the cost f has to pay for the congestion it caused on associated links.

$Q_f = q_f - \sum_{f' \in C_f} q_{f'}$, which means Q_f for flow f is equal to the relay cost paid to its father minus the relay awards obtained from its child flows. Given q_f 's interpretation, we can interpret Q_f as the net expense of flow f while handling data relay relationships.

3.4.4 Convergence Analysis

In the convergence analysis, we make similar assumptions on the problem P with those in the primal-based algorithm. (refer to 3.3.3)

This dual-based algorithm is derived strictly along the dual-based procedure to solve such a convex programming problem. With the assumption that the utility functions U_f are increasing, twice continuously differentiable and strictly concave in the interval X_f, i.e., $-U_f^{'}(x_f) \geq \gamma_f > 0$, we can borrow **Theorem 4** in [5] to use here. The theorem states that as long as the step-size α satisfies some constraint, the

sequence of vectors $\vec{x}^{(n)} = (\vec{x}_f^{(n)}, f \in \mathcal{F})$ converges to the unique optimal solution of problem **P**. Therefore the convergence properties of our dual-based algorithm can be explained in a similar way as in [5].

Chapter 4 Protocol Description and Performance Evaluation

4.1 Motivations

As we can see, our algorithms presented in Chapter 3 treat each flow and link as individual entity which has capabilities of executing many tasks including computing and communicating, etc. For example, in primal-based algorithm, a link should be able to measure its congestion level and transmit it to a set of flows, while a flow has to decide its "flow congestion penalty", "data availability penalty", update its rate and transmit the rate to associated links and flows. None of these exists in current Internet. Furthermore, our work is based on application layer multicast, which can be easily deployed without changing underlying infrastructure. So we cannot assume that links (associated routers) have the additional computing and communicating abilities other than provided by current Internet. Overlay multicast is constructed and maintained by cooperation of participating end hosts, so in our protocol all the operations assumed to be handled by links and flows are pushed to end hosts.

4.2 Protocols

Two algorithms have been proposed in Chapter 3, i.e., primal-based algorithm and dual-based algorithm. The essential difference between them is: in primal-based algorithm rates vary gradually and shadow prices are given as functions of the rates;

while in dual-based algorithm shadow prices vary gradually and rates are given as functions of the prices. Put into overlay layer environment, the protocol we design for these algorithms should be based on the exchange of messages between end hosts, which leads to an equilibrium state of the whole system. Before introducing the protocols, we define some useful notations first.

4.2.1 Notations

In order to push all necessary operations to end hosts, we should find suitable hosts to manage flows and links in above-proposed algorithms. Thus for each flow $f \in \mathcal{F}$ and each link $l \in \mathcal{L}$ we define delegate hosts FH_f and LH_l respectively. Then the optimal resource allocation in overlay multicast is achieved through communicating messages among these hosts.

From link l's algorithm in both algorithms, we can see that l needs to communicate with all flows in $F(l)$. Therefore, LH_l should store a set of hosts $C(l) = \{FH_f : f \in F(l)\}$.

Similarly, from flow f's algorithm in both algorithms, we can see that f needs to communicate with its father flow and children flows. So FH_f should store a set of hosts $R(f) = \{FH_{f'} : f' = \pi_f \; or \quad f' \in C_f \quad\}$. Also, f needs to send its rate to a set of hosts $P(f) = \{LH_l : l \in L(f)\}$ to help them update corresponding link's price information.

4.2.2 Protocol for primal-based algorithm

From above statements, we can see that each delegate host for flows or links need to communicate with sets of other hosts. First we define three kinds of messages circulated in the protocol. Note that in real implementation, these control messages

can be sent as separate packets or conveyed through a field in ordinary data packets.

FRU (Flow Rate Update) : A flow f's delegate host FH_f will send **FRU** messages to hosts in $P(f)$ and $\{FH_{f'} : f' = \pi_f\}$. The message contains these fields: f, x_f.

FRP (Flow Rate & Penalty): A flow f's delegate host FH_f will send **FRP** messages to hosts in $\{FH_{f'} : f' \in C_f\}$. The message contains these fields: f, x_f, s, t.

LCU (Link Congestion Update): A link l's delegate host LH_l will send **LCU** messages to hosts in $C(l)$. The message contains these fields: l, ε_l.

The protocol runs in a time-varying asynchronous network environment, so we cannot take everything as ideal as in algorithms. Take link's algorithm for example, a link l updates its congestion indicator after receiving rates from all flows in $F(l)$. In the protocol, if we force the link delegate not to update until receiving **FRU** messages from all hosts in $C(l)$, deadlock is very likely to happen due to the varying network conditions such as the available bandwidth of links. To ensure a smooth process in the protocol, we create some tables in end hosts to store information collected from these messages and let hosts do updates periodically using these stored information. Therefore in case that some messages cannot arrive in time for the update, we can query suitable table entries to get the information of last round to use. Since it is an iterative process, such alternative will not influence the final result and may only reduce the convergence rate a bit.

For a host FH_f, it has three tables named LT_f (Link Table), FT_f (Father Table), CT_f (Child Table). Their items are illustrated in Table 4.1.

LT_f : totally $\|L(f)\|$ entries, entry i :

i (link i)	ε_i (congestion indication of link i)	LH_i (delegate of link i)

FT_f : only one entry

j (# of π_f)	x_j (rate of π_f)	FH_j (delegate of π_f)	s	t

CT_f : totally $\|\{f' : f' \in C_f\}\|$ entries, entry k :

k (# of f')	x_k (rate of f')	FH_k (delegate of f')

Table 4.1 Tables in flow's delegate host (primal-based)

For a host LH_l, it has only one table named RT_l (Rate Table), whose items are shown in Table 4.2.

RT_l : totally $\|F(l)\|$ entries, entry i :

i (flow i)	x_i (rate of flow i)	FH_i (delegate of flow i)

Table 4.2 Table in link's delegate host (primal-based)

In order to create such tables in end hosts, information such as $F(l)$, $L(f)$, π_f and C_f has to be known at first. In an overlay multicast tree structure, every node has knowledge of its father node and children nodes, so FT and CT are easy to build.

And $L(f)$ can be derived from $F(l)$ if there's a central database to store $F(l)$ for all the links.

So here we choose a centralized approach, where the server collects the physical links information from all flows and then construct the database accordingly. For each flow, we assume that it can get its underlying links information by using existing network tools such as traceroute. And each receiver updates its corresponding flow route information when it joins, leaves the multicast session or its underlying path is changed. We can periodically investigate the path to see whether an update is needed. Considering the case that most routes in current Internet is relatively stable, the update overhead is very small.

So the server will number each flow upon its joining and store its route information. Before the rate control process begins, the server collects all $F(l)$ s, deduce all $L(f)$ s and then number all the links. Then the server assigns delegates for all flows and links and then fills corresponding entries in above tables.

Consider an end host h which is delegate of both flow f and link l. As FH_f, h needs to communicate with hosts in $P(f)$ and $R(f)$. While as LH_l, h needs to communicate with hosts in $C(l)$. So when assigning delegates, the messaging overhead can be reduced if the intersection set among $P(f)$, $R(f)$ and $C(l)$ is maximized by properly choosing FH_f and LH_l. Intuitively, either f's sender or receiver can be designated as FH_f for its uniqueness, and we use receiver in our protocol. While selecting LH_l, we obey two principles: 1) $LH_l \in C(l)$. 2) If l is the only access link of host h to connect to the Internet, also $h \in C(l)$ then $LH_l = h$.

When updating ε_l, link l's delegate LH_l needs to know l's available bandwidth. We assume that the available bandwidth of physical links can be measured by network tools such as pathchar and pathrate, in an end-to-end manner. Since LH_l is also the delegate of some flow $f \in F(l)$, it can use these tools to get available bandwidth for all the links in $L(f)$ although we only need the one for l.

We present the protocol of primal-based algorithm in Figure 4.4. $L(h)$ denotes the set of links that host h delegates and $F(h)$ denotes the set of flows that h delegates.

End host h

On receiving a FRU:

Read f, x_f fields, use f as index, query all CT and RT, update x_f in corresponding entries;

On receiving a FRP:

Read f, x_f, m, n fields, use f as index, query all FT, update x_f, m, n in corresponding entries;

On receiving a LCU:

Read l, ε_l fields, use l as index, query all LT, update ε_l in corresponding entries;

Periodically: (n th iterative step)

For $\forall l \in L(h)$

Consult RT_l, get the rates and delegate hosts for all flows $f \in F(l)$;

Update $\varepsilon_l^{(n)} = \begin{cases} 0 & if \quad \sum_{f \in F(l)} x_f \le c_l \\ 1 & if \quad \sum_{f \in F(l)} x_f > c_l \end{cases}$;

Send LCU to all delegate hosts in RT_l, with the fields l, $\varepsilon_l^{(n)}$;

For $\forall f \in F(h)$

Consult LT_f, FT_f and CT_f;

Compare $x_f^{(n-1)}$ with rates in $CT_f \Rightarrow t' = t$ [(number of rates equal to $x_f^{(n-1)}$ in CT_f)+1];

$$s' = t \sum_{l \in L(f)} \varepsilon_l^{(n-1)} + s;$$

$$P_f^{(n)} = s'/t';$$

Compare $x_f^{(n-1)}$ with the rate $x_{\pi_f}^{(n-1)}$ in FT_f

if $x_f^{(n-1)} > x_{\pi_f}^{(n-1)}$

$$R_f^{(n)} = 1;$$

else

$$R_f^{(n)} = 0$$

Update rate $x_f^{(n)} = \left[x_f^{(n-1)} + \alpha_n U_f'(x_f^{(n-1)}) - \beta_n (P_f^{(n)} + R_f^{(n)}) \right]_{X_f}$

Send FRU to delegates in LT_f and FT_f with fields $f, x_f^{(n)}$;

Send FRP to delegates in CT_f with fields f, $x_f^{(n)}, s', t'$.

Figure 4.4 Protocol of primal-based algorithm

4.2.3 Protocol for dual-based algorithm

In this subsection, we present the protocol for dual-based algorithm. The framework is similar with the above protocol for primal-based algorithm except the different messages and tables.

We define two messages as below.

FRU (Flow Rate Update) : A flow f's delegate host FH_f will send **FRU**

messages to hosts in $P(f)$ and $R(f)$. The message contains these fields: f, x_f.

LPU (Link Price Update): A link l's delegate host LH_l will send **LPU** messages to hosts in $C(l)$. The message contains these fields: l, p_l.

Correspondingly, tables are defined as illustrated in Table 4.3 and Table 4.4.

For a host FH_f, it has two tables named LT_f (Link Table) and RT_f (Relative Table), Their items are illustrated in Tab.3.

LT_f : totally $\|L(f)\|$ entries, entry i :

i (link i)	p_i (link price of link i)	LH_i (delegate of link i)

RT_f : totally $\|R(f)\|$ entries, entry j

j (flow j)	x_j (rate of flow j)	FH_j (delegate of flow j)

Table 4.3 Tables in flow's delegate host (dual-based)

For a host LH_l, it has only one table named FT_l (Rate Table), whose items are shown in Tab.4.

FT_l : totally $\|F(l)\|$ entries, entry i :

i (flow i)	x_i (rate of flow i)	FH_i (delegate of flow i)

Table 4.4 Table in link's delegate host (dual-based)

We present the protocol of dual-based algorithm in Figure 4.5.

End host h

On receiving a FRU:

Read f, x_f fields, use f as index, query all FT and RT, update x_f in corresponding entries;

On receiving a LPU:

Read l, p_l fields, use l as index, query all LT, update p_l in corresponding entries;

Periodically: (n th iterative step)

For $\forall l \in L(h)$

Consult FT_l, get the rates and delegate hosts for all flows $f \in F(l)$;

Update link price $p_l^{(n)} = [p_l^{(n-1)} + \alpha(\sum_{f \in F(l)} x_f^{(n-1)} - c_l)]_+$;

Send LPU to all delegate hosts in FT_l, with the fields l, $p_l^{(n)}$;

For $\forall f \in F(h)$

Consult LT_f, get the link prices and delegate hosts for all links $l \in L(f)$;

Update flow price $P_f^{(n)} = \sum_{l \in L(f)} p_l^{(n)}$;

Consult RT_f, get the rates and delegate hosts for f's father and children flows;

Update relay price $q_f^{(n)} = [q_f^{(n-1)} + \alpha(x_f^{(n-1)} - x_{\pi_f}^{(n-1)})]_+$;

Update $q_{f'}^{(n)}$ for each $f' \in C_f$: $q_{f'}^{(n)} = [q_{f'}^{(n-1)} + \alpha(x_{f'}^{(n-1)} - x_f^{(n-1)})]_+$;

Update data price $Q_f^{(n+1)} = q_f^{(n+1)} - \sum_{f' \in C_f} q_{f'}^{(n+1)}$;

Update rate $x_f^{(n)} = [U_f'^{(-1)}(P_f^{(n)} + Q_f^{(n)})]_{X_f}$

Send FRU to delegates in LT_f and RT_f with fields $f, x_f^{(n)}$.

Figure 4.5 Protocol of dual-based algorithm

4.3 Performance Evaluation

In Chapter 3, we have explained the convergence properties of the algorithms. Next we will study our protocols' performance, including convergence properties and various overheads, through simulation experiments carried out in an asynchronous time-varying environment. Also, we will evaluate primal-based and dual-based protocols comparatively to show their pros and cons.

4.3.1 Simulation Setup

We use the Boston BRITE topology generator to setup our experimental network. BRITE provides eight different generation models, and here we choose the top-down hierarchical topology model. The two-level hierarchical topologies are in accordance to the two level routing hierarchy that has persisted in the Internet since ARPANET evolved into a network of networks interconnecting multiple autonomous systems. Figure 4.6 depicts the structure of the hierarchical topology we use in simulation.

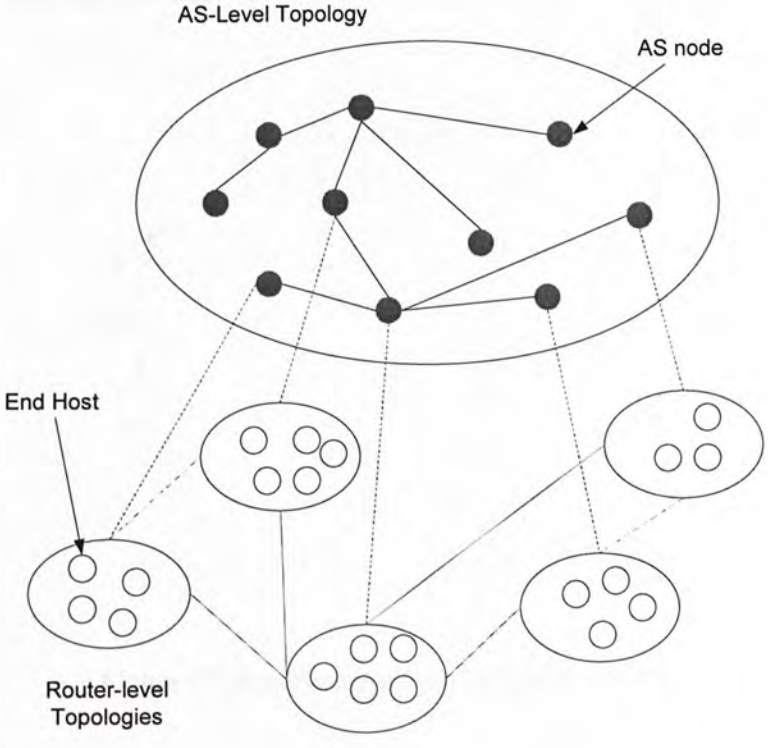


Figure 4.6 Simulation Topology

We set 10 nodes in the AS-level topology and each AS node corresponds to a router-level topology consisting of 100 nodes. So there are totally 1000 nodes in our simulation. In an overlay multicast session, each overlay node in the tree is an end host connected to a single router. The bandwidths of all links are uniformly distributed 10 and 100Mbps.

We run a single overlay multicast session on this experimental network. The multicast tree is shown in Figure 4.7. The number besides each overlay link denotes corresponding flow number. The tree is constructed as follows. We randomly select 10 router-level nodes delegating connected end hosts and assign one of them as the server node (node 0). For all the 10 nodes, node degree (maximum number of children it can have) is set to 4. Initially, node 1 joins the session and attaches to node 0 in the overlay multicast tree. Note that we use the most general shortest-path routing in finding paths. In every 42 seconds thereafter, a new node joins the session. Each new node attaches itself to one of the existing multicast nodes which is closest to the new node in terms of hop numbers. Certainly, during the process, the node degree constraints cannot be violated.

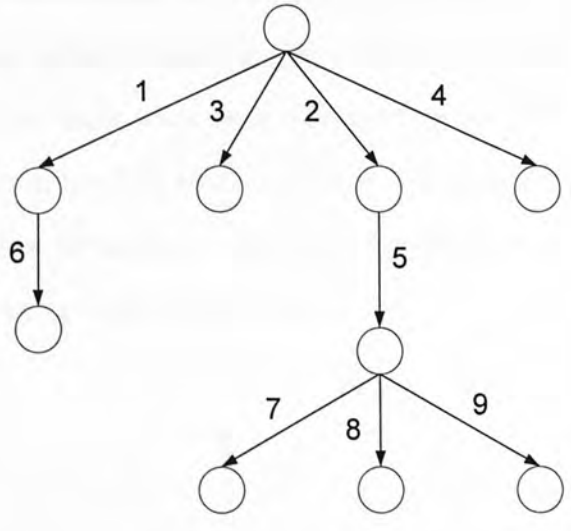


Figure 4.7 Experimental Overlay Multicast Tree

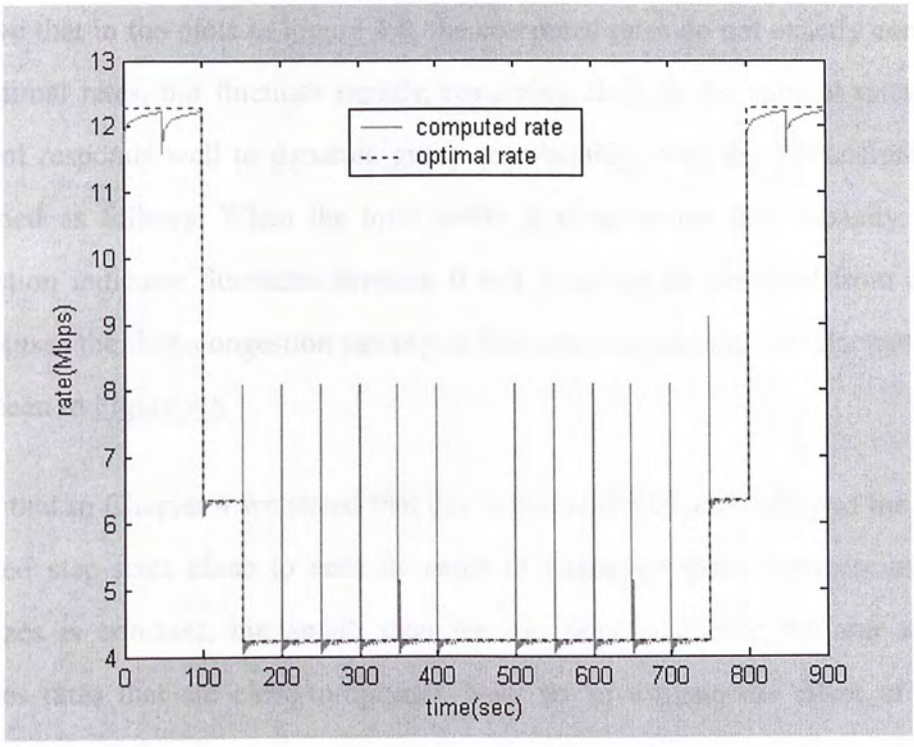
In our simulations, for each flow f, we let its utility function $U_f(x_f)$ to be

$\ln(x_f)$ with its minimum and maximum rates being 1Mbps and 42Mbps. Taking various factors into account, including trans-coding techniques, computing complexities and communicating overhead in end hosts, we let each flow's delegate host update the flow's rate every 1 second in all the simulations. What's more, we will refer to primal-based protocol as Protocol A and dual-based protocol as Protocol B in later sections for brevity.

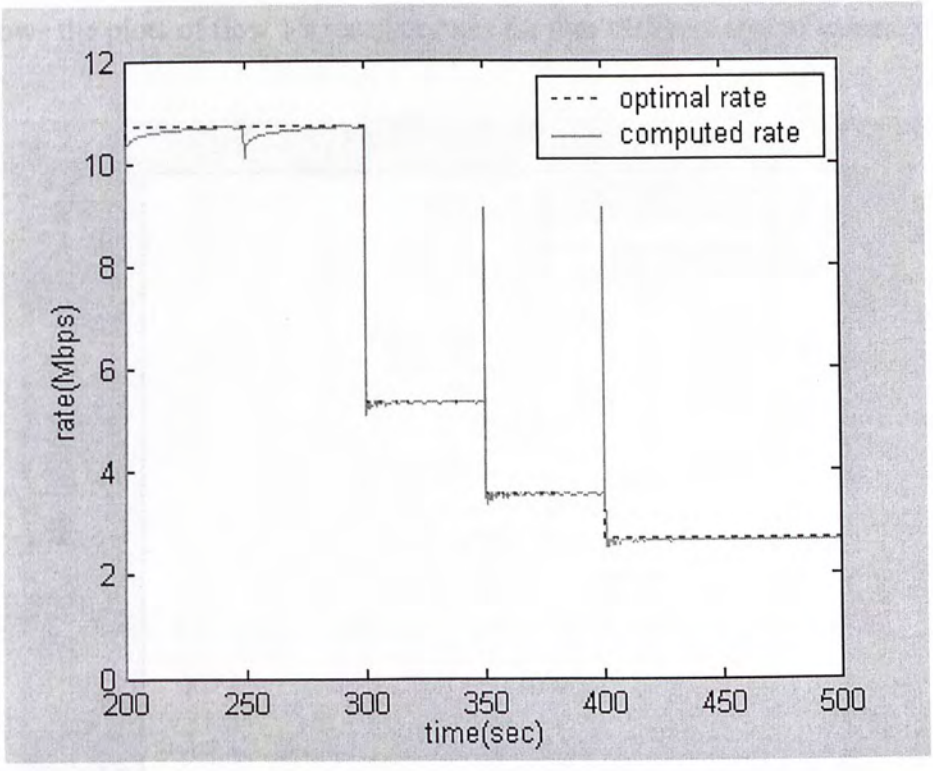
4.3.2 Rate Convergence Properties

Protocol A:

According to the constraints (4)-(6) for step-sizes α_n and β_n, if not specified, we will use $\alpha_n = \frac{1}{n}$, $\beta_n = \frac{1}{\sqrt{n}}$ in our simulation as default. Figure 4.8(a) and 4.8(b) show the (achieved) receiver rates of flow 1 and flow 5 along with the theoretical optimal rates (these two flows are chosen arbitrarily, and the curves of other receiver rates also exhibit a similar trend). The rates are plotted every 1 second, which is also the time interval between successive rate updates at end hosts. Note that the sudden changes in the optimal rates in 8(a) and 8(b) are due to the arrival or departure of some flows. These flows share some links with the flow we plot, so when they arrive or depart, the congestion level on these links will change and thus cause changes on plotted rate. The figures demonstrate that the computed flow rates track the optimal rates closely even as the optimal rates change.



(a) Flow 1

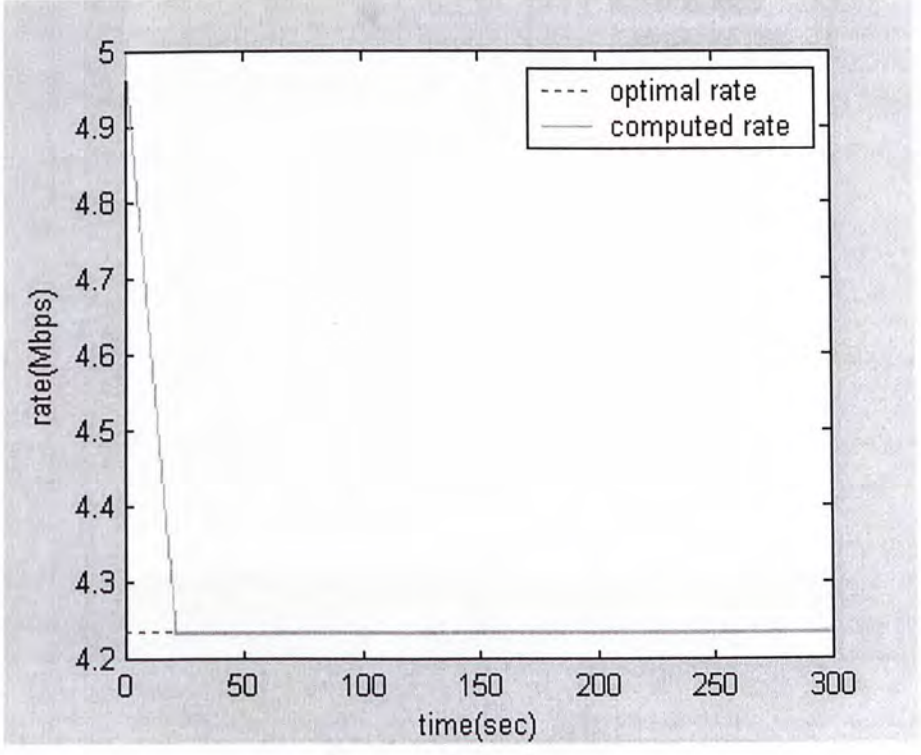


(b) Flow 5

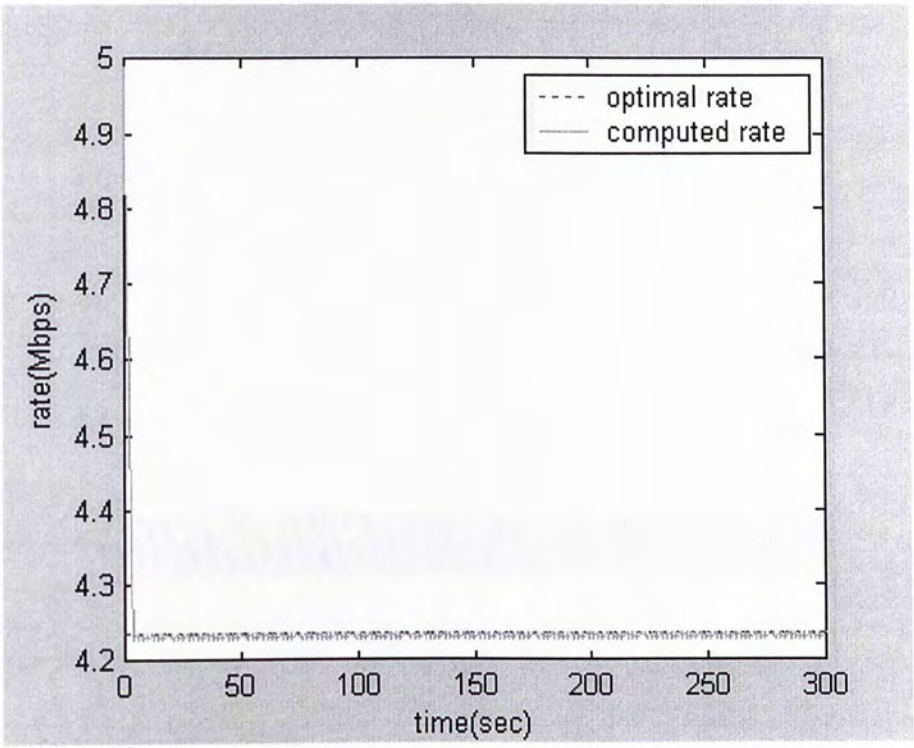
Figure 4.8 Convergence of Flow rates (Protocol A)

Observe that in the plots in Figure 4.8, the computed rates do not exactly converge to the optimal rates, but fluctuate rapidly, remaining close to the optimal rates. So our protocol responds well to dynamic group membership. And the fluctuations can be explained as follows. When the total traffic is close to the link capacity, the link congestion indicator fluctuates between 0 and 1, as can be expected from intuition. This causes the flow congestion penalty to fluctuate too, causing rate fluctuations like those seen in Figure 4.8.

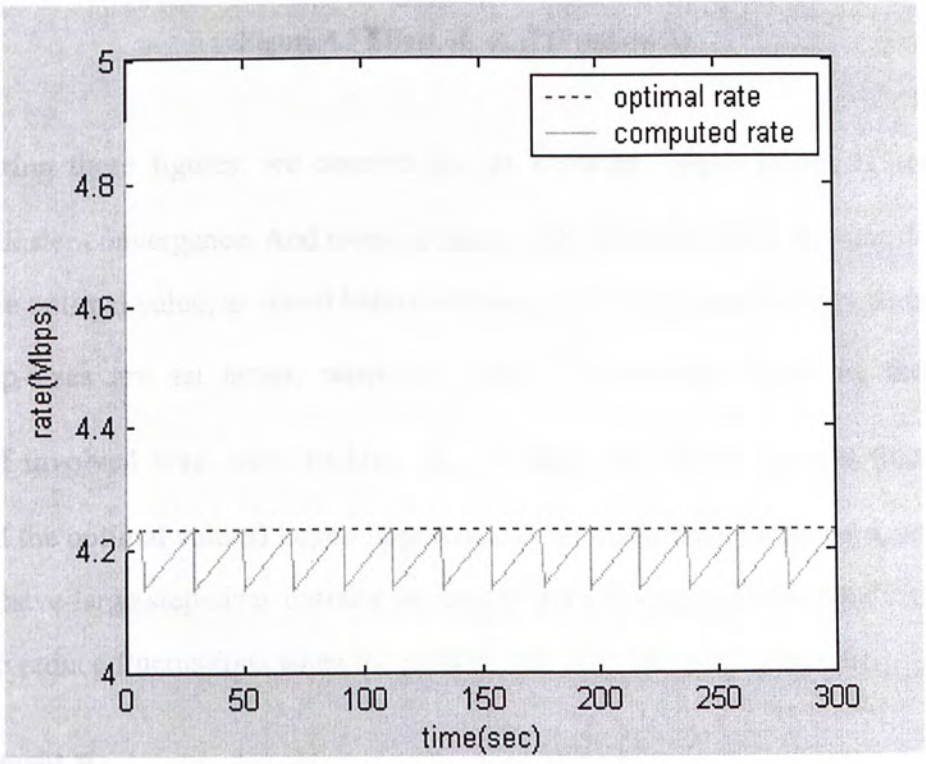
Recall that in Chapter 4 we stated that due to the non-differentiability of the problem we need step-sizes close to zero in order to guarantee exact convergence. If the step-sizes is constant, but small, then we can only guarantee that our algorithm achieves rates that are close-to-optimal. Now we investigate the effect of constant step-sizes $\alpha_n = \alpha$, $\beta_n = \beta$. As we discussed in Chapter 3, constant step-sizes can also lead to convergence results, but not as precise as diminishing step-sizes. Figure 4.9 shows the plots of flow 1's receiver rates for four different sets of values of α, β.



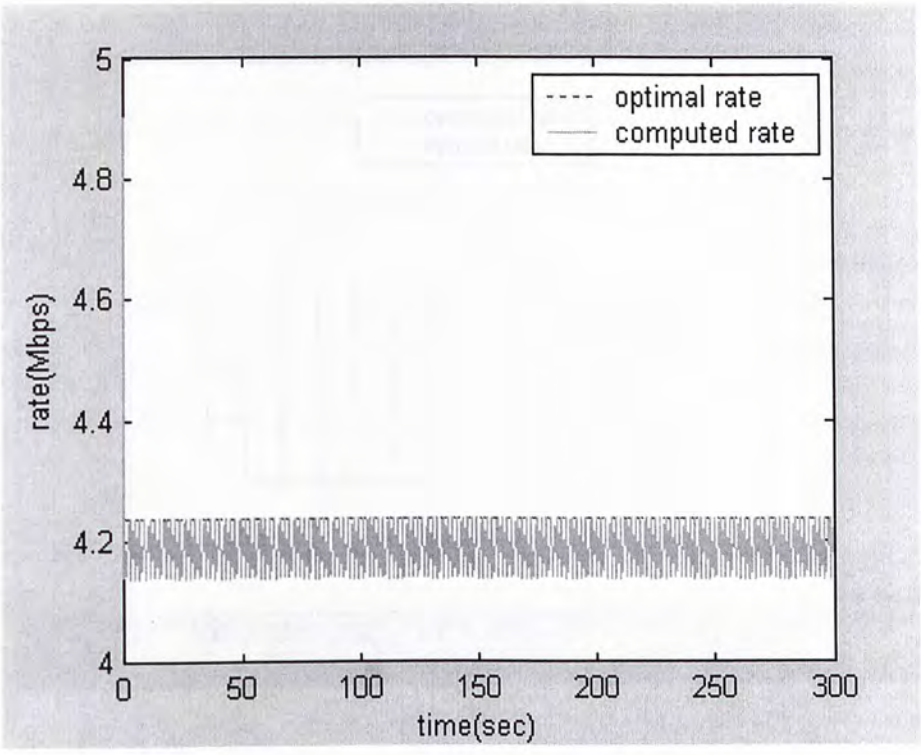
(a) $\alpha = 0.001, \beta = 0.002$



(b) $\alpha = 0.001, \beta = 0.01$



(c) $\alpha = 0.001, \beta = 0.1$



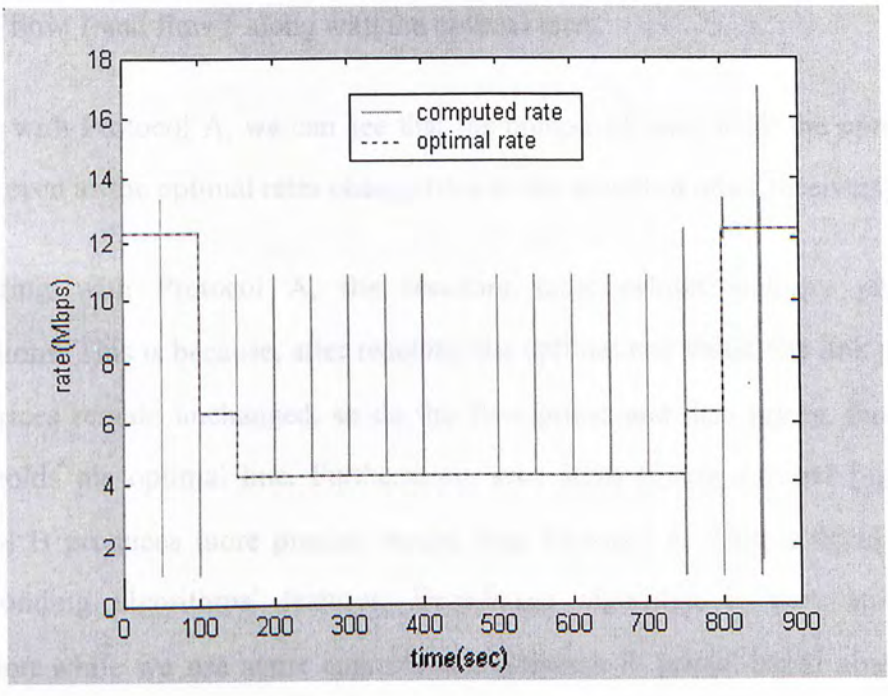
(d) $\alpha = 0.01, \beta = 0.1$

Figure 4.9 Effect of α, β (Protocol A)

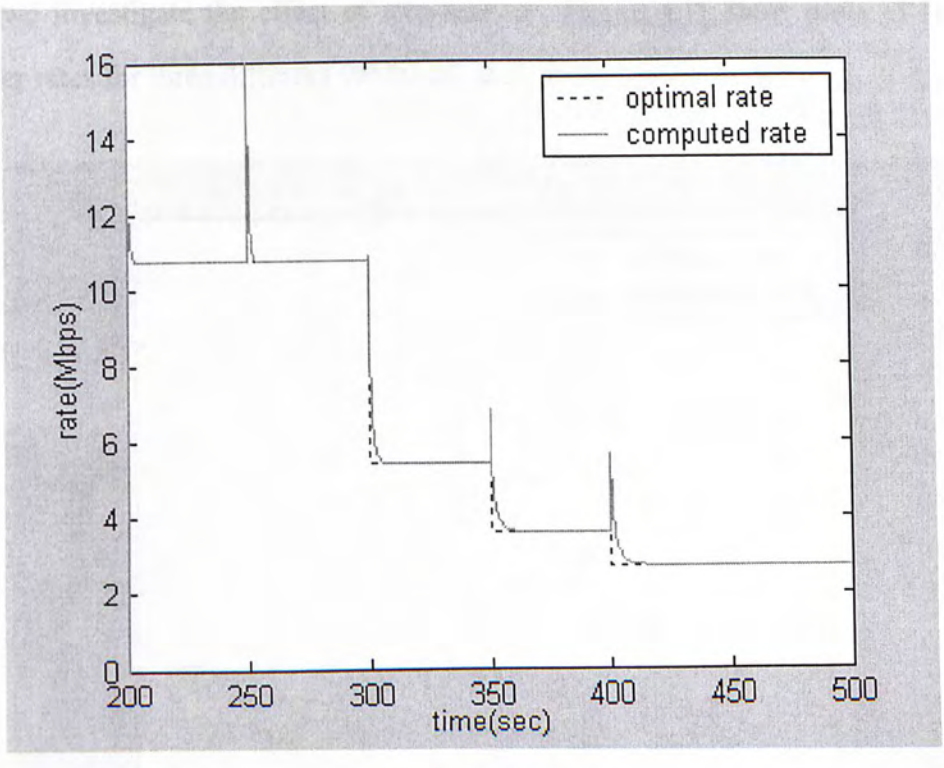
Comparing these figures, we observe that as expected, larger values of α and β lead to faster convergence. And using constant step-sizes may result in some deviation from the optimal value, as stated before in theory. The deviation becomes worse when the step-sizes are set larger, especially when β increases. However, there is a tradeoff involved here, since making α, β large also makes the rate fluctuations (around the optimal values) larger. In practice, if we use constant step-sizes, we would like to have large step-sizes initially (to ensure fast convergence) and small step-sizes later (to reduce fluctuations when the rates are close to the optimal values).

B. Protocol B

If not specified, we will use $\alpha = 0.0005$ in our simulation as default. All the settings are similar with those in Protocol A. Figure 4.10(a) and 4.10(b) show the receiver



(a) Flow 1



(b) Flow 5

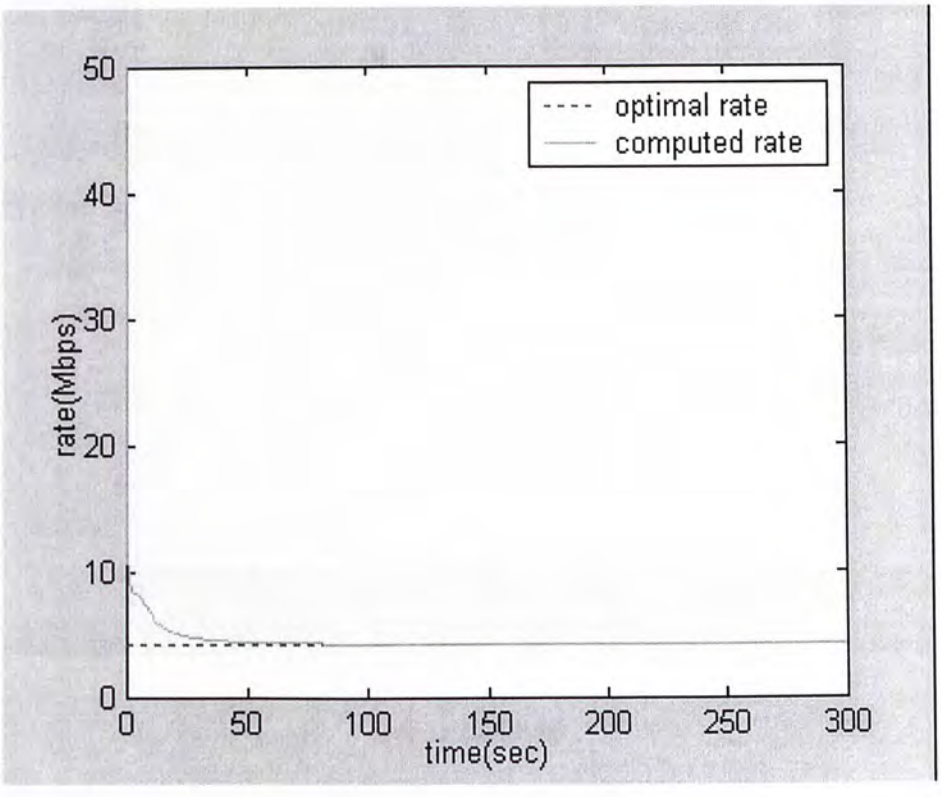
Figure 4.10 Convergence of Flow Rates (Protocol B)

rates of flow 1 and flow 5 along with the optimal rates.

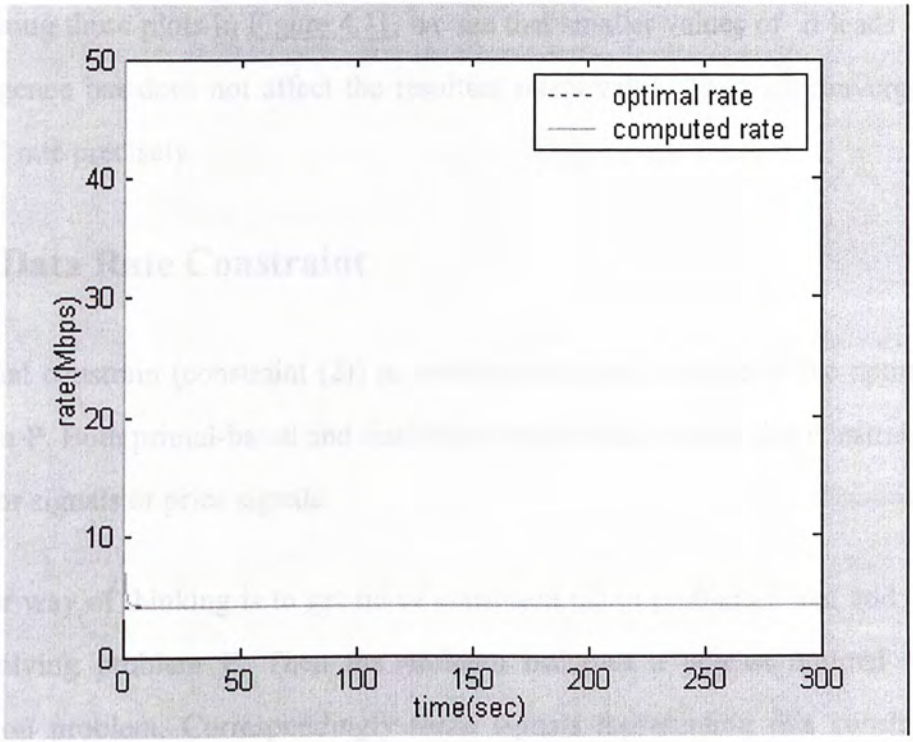
Similar with Protocol A, we can see that the computed rates track the optimal rates closely even as the optimal rates change (due to the arrival of other receivers).

Comparing with Protocol A, the resultant rates exhibit stability rather than fluctuations. This is because, after reaching the optimal rate value, the link prices and relay prices remain unchanged, so do the flow prices and data prices, thus the rate value holds the optimal line. Furthermore, seen from Figure 4.8 and Figure 4.10, Protocol B produces more precise results than Protocol A. This is decided by the corresponding algorithms' features. Dual-based algorithm is very strict in its derivation while we use some approximate elements in primal-based algorithm for simplicity reasons, e.g., the R_f's part in x_f's computation.

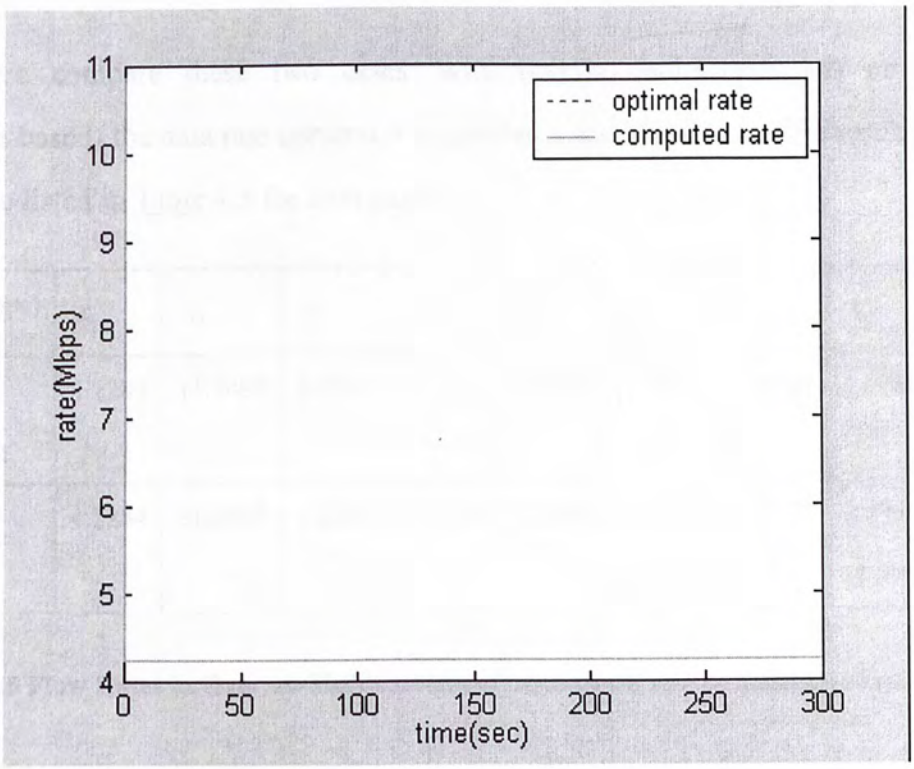
Then we investigate the effect of step-size α. Figure 4.11 show plots of flow 1's receiver rates for three different values of α.



(a) α =0.005



(b) α =0.0005



(c) α =0.00005

Figure 4.11 Effect of α (Protocol B)

Comparing these plots in Figure 4.11, we see that smaller values of α leads to faster convergence but does not affect the resultant rate's value. They all converge to the optimal rate precisely.

4.3.3 Data Rate Constraint

A special constrain (constraint (2)) in overlay multicast is added to the optimization problem **P**. Both primal-based and dual-based algorithms include this constraint using indicator signals or price signals.

Another way of thinking is to get rid of constraint (2) in problem P and add it at last after solving problem **P**. Then the problem becomes a unicast optimal resource allocation problem. Correspondingly those signals representing this constraint are removed from both algorithms. The resultant flow rates are then adjusted so that their rates are no higher than their father flow rates.

Here we compare these two cases: with (overlay-multicast-based) or without (unicast-based) the data rate constraint in problems and algorithms. The resultant flow rates are listed in Table 4.5 for both cases.

Rate(Mbps)	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9
Overlay multicast	4.2282	11.3407	4.2282	4.2282	2.6788	11.3407	2.6788	2.6788	2.6788
Unicast	4.2284	11.3407	4.2285	4.2278	2.6787	14.3227 (4.2284)	2.6787	2.6792 (2.6787)	2.6787

Table 4.5 Flow Rates in Overlay-Multicast-based mechanism and Unicast-based mechanism

The aggregate utilities in overlay-multicast-based mechanism is $\sum_{i=1}^{9} \ln(x_i) = 13.1434$.

In unicast-based mechanism, after adjusting the rates according to constraint (2),

$\sum_{i=1}^{9} \ln(x_i) = 12.1515$. We can see that unicast-based mechanism is sub-optimal to overlay-multicast-based mechanism in maximizing aggregate utilities, which is also the objective of problem P. So we discuss overlay-multicast-based protocols in our thesis.

4.3.4 Link Measurement Overhead

In both Protocol A and Protocol B, involved links' available bandwidths need to be measured periodically. Such tasks are put on delegate end hosts. Now we discuss the overhead of these tasks. We plot the number of links a multicast tree session contains in Figure 4.12 with varied d. Here d refers to the node degree constraint in tree building. We investigate the effect of three different d: $d=2$; $d=10$; $d=\infty$.

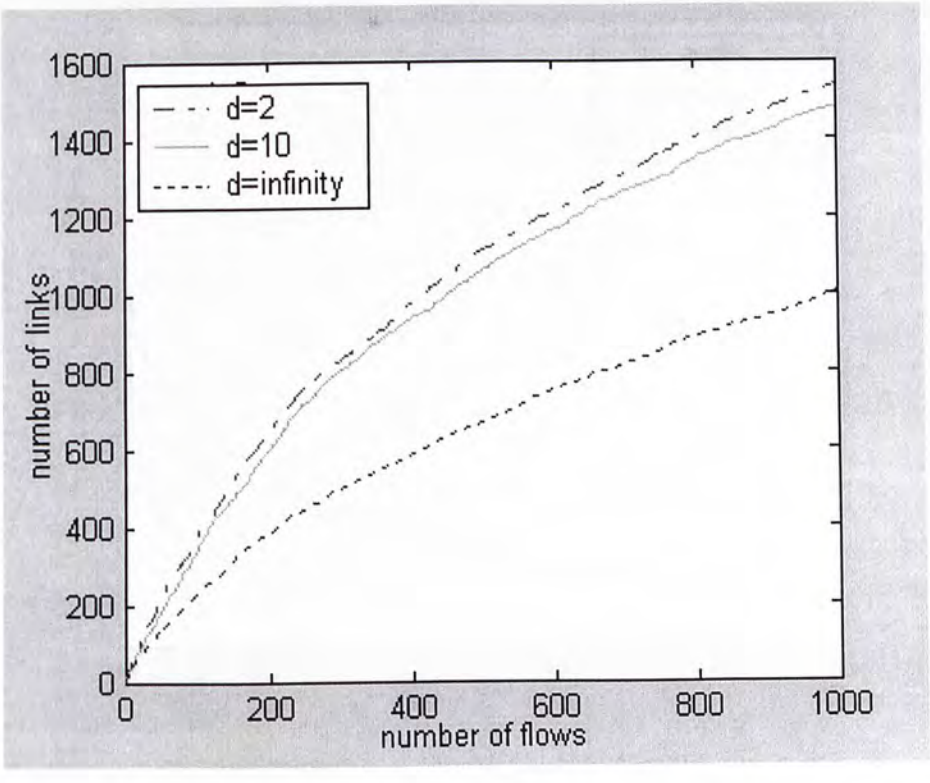


Figure 4.12 Link Measurement Overhead

Figure 4.12 shows that the number of links grows sub-linearly along with the expansion of the multicast session. The reason can be explained as follows. Given a

fixed amount of network links, more links are shared by different flows when more and more flows join in the session. It is also shown that the number of links decrease along with the increase of d. This is because some receivers cannot be child nodes of their closest neighbor nodes in session because of the degree constraint. When the degree constraint is relaxed, these receivers can then stream from the closest neighbors, thus the lengths (number of links) of corresponding flows can be decreased.

We use receiver-based scheme in our work. So each flow has its own delegate host and the number of receivers is equal to the number of flows in the multicast session. Figure 4.13 shows the number of links per flow in a session, which can be also considered as the link measurement overhead per host.

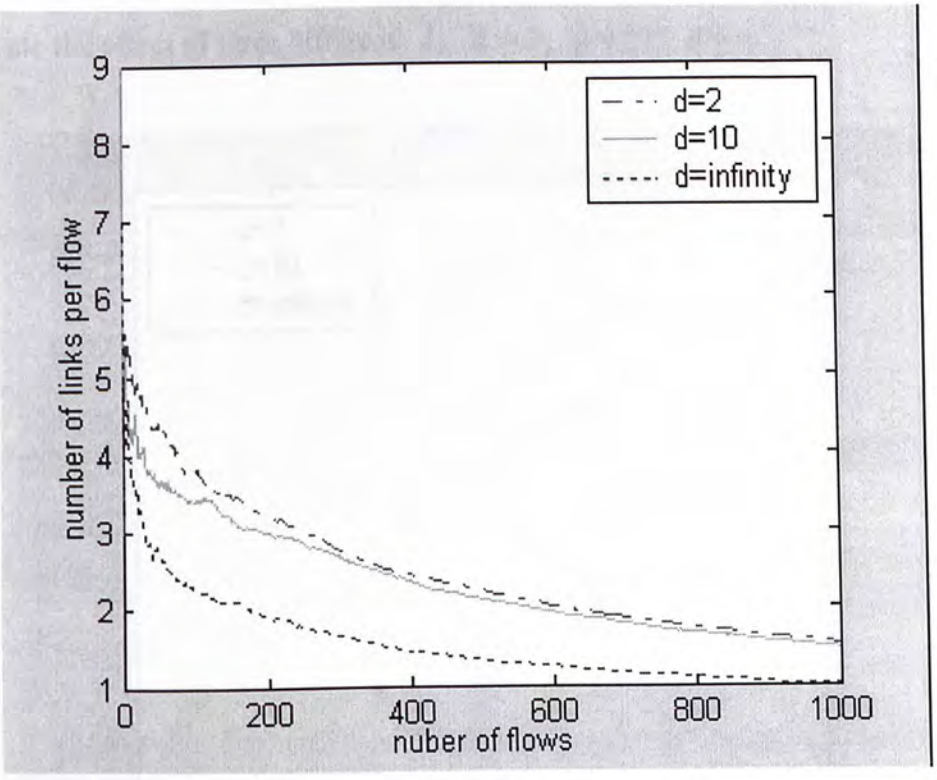


Figure 4.13 Link Measurement Overhead Per Flow

Figure 4.13 shows that the number of links per flow decreases with the increase of session size. This is consistent with the convexity of the plots in Figure 4.12. This

shows our protocols' scalability in link measurement overhead. Similar with Figure 4.12 , larger d , smaller number of links per flow.

4.3.5 Communication Overhead

Both protocol A and protocol B uses communicating messages between end hosts to achieve the global equilibrium, so the communication overhead is another concern of us. As we see in Figure 4.9 and Figure 4.11, different selected step-sizes may influence the convergence rate of the protocols. Thus we have no knowledge about how many iteration steps are needed before reaching the equilibrium. So we investigate the communication overhead, in other words, the number of messages communicated, in one iteration step here. Figure 4.14 plots the number of messages in one iteration along with the number of flows in a session. Similar with 4.3.4, we investigate the effect of three different d: $d=2$; $d=10$; $d=\infty$

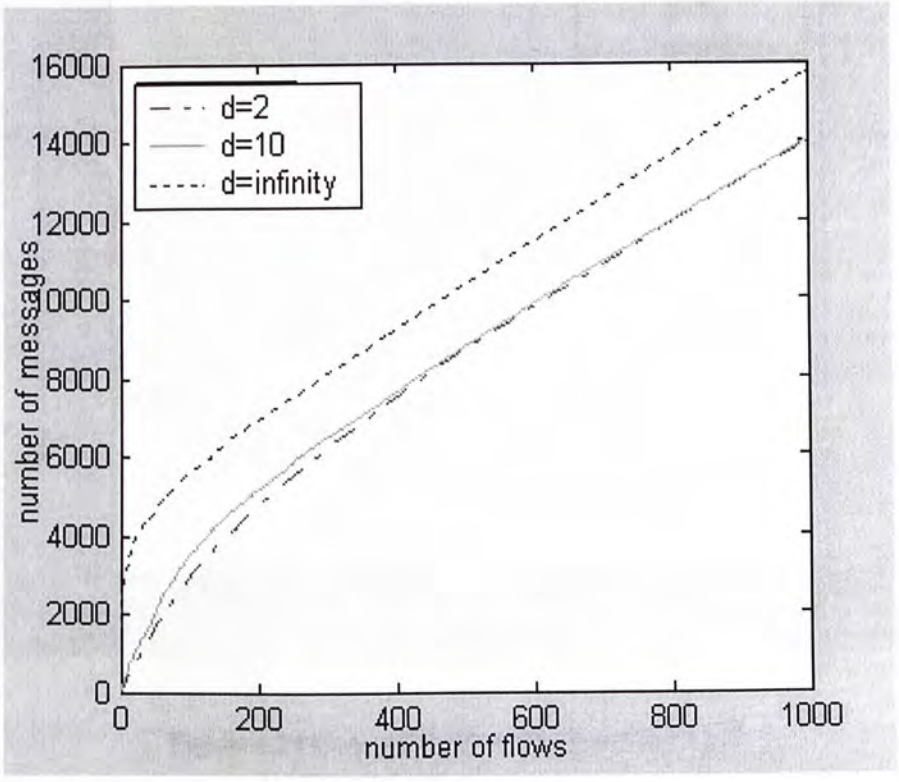


Figure 4.14 Communication Overhead

Figure 4.14 shows that increasing d results in more messages in one iteration. This

can be explained as follows. Increasing d can reduce the number of links in a session, which results in less link update messages. However, as d increases, more and more flows start to share some links, especially those near to the sender node. As our protocols describe, exchanging messages are needed among the flows that share a single link. We use receiver-based scheme, so different flow has different delegate host. From this observation, increasing d will result in more communicating messages among these hosts. In summary, increasing d has two opposite impact on resultant number of messages. And Figure 4.14 shows that the negative effect gains the leading role.

Similar with 4.3.4, Figure 4.15 shows the number of messages per flow in an iteration, which can be also considered as the communication overhead per host.

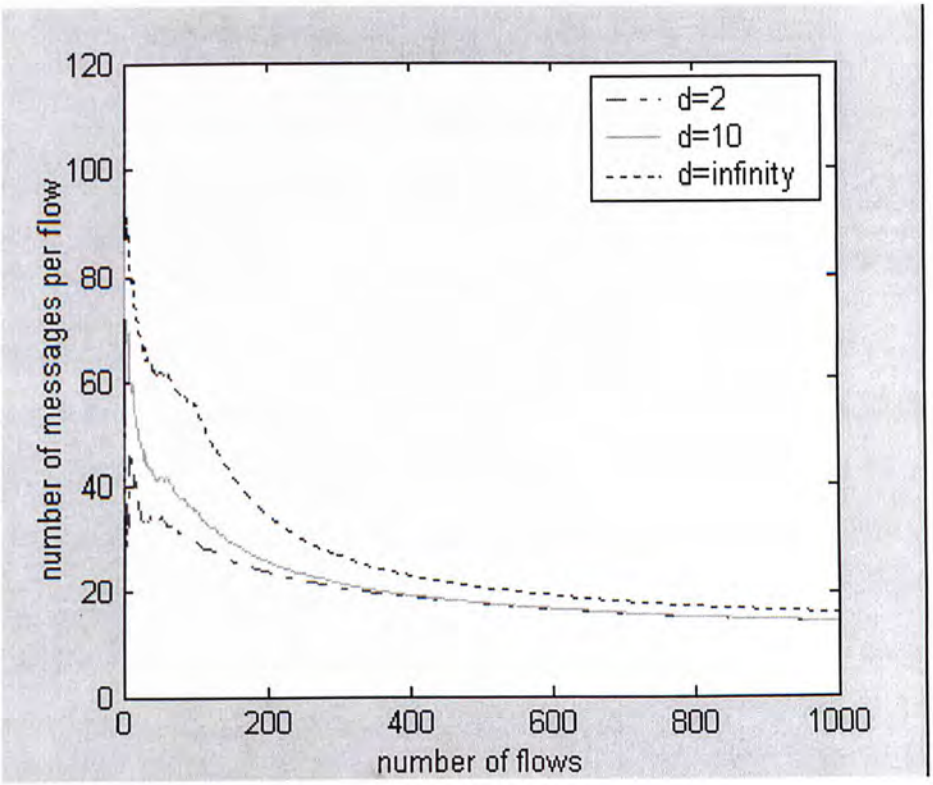


Figure 4.15 Communication Overhead Per Flow

Figure 4.15 shows that the number of messages per flow decreases with the increase of session size and tends to be stable in large sessions. This again shows our

protocols' scalability in the aspect of communication overhead.

Chapter 5 Conclusion Remarks and Future Work

This thesis proposes a new paradigm for solving the problem of congestion control in multicast service in Internet with the use of overlay networks and addresses some design issues. We propose solution approaches to address the new challenges in overlay multicast rate control problem, which make the solutions totally different from previous ones, although using similar approaches. We then implement these approaches in protocols based on coordination of end hosts. Our simulations show the scalability and efficiency of our protocols.

There are several related issues that need to be investigated further. Note that all the convergence results presented in this thesis are for synchronous updates. Although the algorithms converge to the optimal rates in all of our simulations carried out in asynchronous environments, derivation of a formal proof of convergence for that case is an interesting problem from the theoretical point of view. Also note that in the algorithms, we use a server node to collect the route information for all flows, however, other distributed mechanisms will also work here, which we consider complementary work to this thesis.

References

[1] Bo Hong, Viktor K. Prasanna, "Bandwidth-Aware Resource Allocation for Heterogeneous Computing Systems to Maximize Throughput", *IEEE International Conference on Parallel Processing*, Taiwan, October 06-09, 2003.

[2] D. F. Ferguson, "The Application of Microeconomics to the design of resource allocation and control algorithms in Distributed Systems", Phd thesis, Columbia University, New York, 1989.

[3] D. F. Ferguson, C. Nikolaou and Y. Yemini, "An Economy for Flow Control in Computer Networks", *Proceedings of the INFOCOM'90*.

[4] J. Sairamesh, D. Ferguson, and Y. Yemini, "An Approach to Pricing, Optimal Allocation and Quality of Service Provisioning in High Speed Packet Networks", *Proceedings of the INFOCOM'95*.

[5] K. Kar, S. Sarkar and L. Tassiulas, "Optimization based rate control for multirate multicast sessions", in *IEEE INFOCOM*, 2001.

[6] K. Kar, S. Sarkar, L. Tassiulas, "A Low-Overhead Rate Control Algorithm for Maximizing Aggregate Receiver Utility for Multirate Multicast Sessions", *Proceedings of SPIE-ITCOM 2001*.

[7] S. Low, D. E. Lapsley, "Optimization Flow Control, I: Basic Algorithm and Convergence", *IEEE/ACM Transactions on Networking*, vol. 7, no. 6, December 1999.

[8] K. Kar, S. Sarkar, L. Tassiulas, "A Simple Rate Control Algorithm for Maximizing Total User Utility", *Proceedings of INFOCOM 2001*, USA, April 2001.

[9] S. McCanne, V. Jacobson, M. Vetterli, "Receiver-Driven Layered Multicast", *Proceedings of ACM Sigcomm'96*, Stanford, CA, September 1996.

[10] X. Li, S. Paul and M. Ammar, "Layered Video Multicast with Retransmission (LVMR): Evaluation of Hierarchical rate control", *Proceedings of IEEE INFOCOM'98*, March 1998.

[11] T. Bially, B. Gold, and S. Seneff, "A technique for Adaptive Voice Flow Control in Integrated Packet Networks", *IEEE Transactions on Communications*, vol. 28, No. 3, March 1980.

[12] T. Turletti and J. C. Bolot, "Issues with multicast video distribution in heterogeneous packet networks", *Packet Video Workshop*, 1994.

[13] F. Kishino, K. Manabe, Y. Hayashi and H. Yasuda, "Variable Bit-Rate Coding of Video Signals for ATM Networks", *IEEE Journal on Selected Areas In Communications*, Vol. 7, No. 5, June 1989.

[14] J. Jannotti, D. K. Gifford, K. L. Johnson, M. F. Kaashoek, and J. W. O. Jr., "Overcast: Reliable multicasting with an overlay network," in *5th Symposium on Operating System Design and Implementation (OSDI)*, Dec. 2000.

[15] Y. Chu, S.G. Rao, and H. Zhang, "A Case for End System Multicast", ACM Sigmetrics 2000, Santa Clara, California, USA, 2000.

[16] P. Francis, " Yoid: Extending the Multicast Internet Architecture", 1999, White paper http://www.aciri.org/yoid/.

[17] D. Pendarakis, S. Shi, D. Verma, and M. Waldvogel, " ALMI: An Application Level Multicast Infrastructure", *Proc. of 3rd Usenix Symposium on Internet Technologies & Systems*, March 2001.

[18] Y. Chawathe, *Scattercast: An Architecture for Internet Broadcast Distribution as an Infrastructure Service*. PhD thesis, Department of EECS, UC Berkeley, Dec. 2000.

[19] C. Diot et al., " Deployment Issues for the IP Multicast Service and Architecture", *IEEE Network,* Jan. 2000, pp. 78C88.

[20] D. P. Bertsekas, *Nonlinear Programming*, Athena Scientific, 1995.

[21] S. Kunniyur, R. Srikant, "End-to-End Congestion Control Schemes: Utility Functions, Random Losses and ECN Marks", *Proceedings of Infocom 2000*, March 2000.

[22] T. M. Stoenescu, M. Liu and D. Teneketzis, "An approach to rate allocation in

multicast", to appear in *IEEE Conference on Decision and Control (CDC)*, December 2003, Maui, Hawaii.

[23] D. Rubenstein, J. Kurose and D. Towsley. The Impact of Multicast Layering on Network Fairness, *Proceedings of ACM Sigcomm'99,* Cambridge, MA, September, 1999.

[24] S. Sarkar and L. Tassiulas: Fair Allocation of Utilities in Multi-rate Multicast Networks, *Proceedings of the 37th Annual Allerton Conference on Communication, Control and Computing*, 1999.

[25] S. Sarkar and L. Tassiulas, Fair Allocation of Discrete Bandwidth Layers in Multicast Networks, *Proceedings of IEEE Infocom 2000*, March 2000.

[26] K. Kar, S. Sarkar, L. Tassiulas, "A Primal Algorithm for Optimization Based Rate Control for Unicast Sessions", Technical Research Report (ISR T. R. 2000-22).

[27] B. T. Poljak, "A General Method of Solving Extremum Problems", *Soviet Math Doklady*, vol.8, no.3, 1967, pp.593-597.

[28] N. Z. Shor, *Minimization Methods for Non-differentiable Functions,* Springer-Verlag, 1985.

[29] J. Widmer, R. Denda, and M. Mauve, "A Survey of TCP-Friendly Congestion Control," *IEEE Network*, May/Jun 2001.

[30] A. Basu and S. J. Golestani, "Architectural Issues for Multicast CongestionControl," *Proc. of NOSSDAV*, 1999.

[31] K. Obraczka, "Multicast Transport Protocols: A Survey and Taxonomy," *IEEE Commun. Mag.*, Jan 1998.

[32] S. J. Golestani and K. K. Sabnani, "Fundamental Observations on Multicast Congestion Control in the Internet," *Proc. of IEEE INFOCOM*, 1999.

[33] D.Chiu and R. Jain, "Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks," *Computer Networks and ISDN Systems*, 1989, pp. 17:1–14.

[34] M.Ammar, S. Y. Cheung, and X. Li, "On the Use of Destination Set Grouping to Improve Fairness in Multicast Video Distribution," Tech-Report: GIT-CC-95-25, Georgia Institute of Technology, Jul 1995.

[35] A. Matrawy, I. Lambadaris, and C. Huang, "Comparison of the Use of Different ECN Techniques for IP Multicast Congestion Control," *Proc. of IEEE ECUMN*, 2002.

[36] A. Matrawy and I. Lambadaris, "Real-Time Transport for Assured Forwarding:An Architecture for both Unicast and Multicast Applications," *Proc. of IEEE ICC*, 2003.

[37] F. Kelly, A. Maulloo, D. Tan, "Rate Control for Communication Networks: Shadow Prices, Proportional Fairness and Stability", *Journal of Operations Research Society,* vol. 49, no. 3, 1998, pp. 237-252.

[38] J-C. Bolot, T. Turletti, and I. Wakeman, "Scalable Feedback Control for Multicast Video Distribution in the Internet," *Proc. of ACM SIGCOMM*, Oct 1994.

[39] T. Turletti and C. Huitema, "Videoconferencing on the Internet," *IEEE/ACM Trans. on Networking*, vol. 4, no. 3, June 1996, pp. 340–51.

[40] L. Rizzo, "pgmcc: A TCP-Friendly Single-Rate Multicast Congestion Control Scheme," *Proc. of SIGCOMM*, 2000.

[41] J. Widmer and M. Handley, "Extending Equation-Based Congestion Control to Multicast Applications," *Proc. of SIGCOMM*, 2001.

[42] K. Shapiro, D. Towsley, and J. Kurose, "Optimization-Based Congestion Control for Multicast Communications," *IEEE Commun. Mag.*, Sep 2002.

[43] F. P. Kelly, "Charging and Rate Control for Elastic Traffic", *European Transactions on Telecommunications*, vol. 8, no. 1, 1997, pp.33-37.

[44] S. Athuraliya, S. Low, D. Lapsley, "Random Early Marking", Submitted for publication, www.ee.mu.oz.au/staff/slow/research/.

[45] L. Vicisano, J. Crowcroft, and L. Rizzo, "TCP-like Congestion Control for Layered Multicast Data Transfer," *Proc. of INFOCOM*, 1998.

[46] A. Legout and E. Biersack, "PLM: Fast Convergence for Cumulative Layered Multicast Transmission Schemes," *Proc. of SIGMETRICS*, 2000, pp. 13–22.

[47] M. Luby, L. Vicisano, and T. Speakman, "Heterogeneous Multicast Congestion Control Based on Router Packet Filtering," presentation at RMRG meeting, Pisa, Italy, Jun 1999.

[48] A. Clerget, "TUF: A Tag-Based UDP Multicast Flow Control Protocol," INRIA technical report 3728, Jul 1999.

[49] A. Darcinschi and S. Fdida, "Efficient Congestion Avoidance Mechanism," in *Proc. of IEEE LCN*, Nov 2000.

[50] A. Matrawy, I. Lambadaris, and C. Huang, "Multicasting of Adaptively Encoded MPEG4 over QoS-Aware IP Networks," *Proc. of IEEE ICC*, 2002.

[51] R. Gopalakrishnan *et al.*, "A Simple Loss Differentiation Approach to Layered Multicast," *Proc. of IEEE INFOCOM*, 2000.

[60] K. Nakauchi, H. Morikawa, and T. Aoyama, "A Network-Supported Approach to Layered Multicast," *Proc. of IEEE ICC*, 2001.

[61] Z. Zhang and V. O. K. Li, "Router-Assisted Layered Multicast," *Proc. of IEEE ICC*, 2002.